UNIVERSIDAD NACIONAL AUTONOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES
ARAGÓN

ESTUDIO DE UN SISTEMA ROBÓTICO TELEOPERADO USANDO LOS PROTOCOLOS ETHERNET Y WI–FI EN REDES LAN Y WAN

T E S I S

QUE PARA OBTENER EL TÍTULO DE

INGENIERO ELÉCTRICO ELECTRÓNICO

P R E S E N T A:

ISMAEL ANGEL SALAZAR RUÍZ

DIRECTOR DE TESIS:
DR. OCTAVIO DÍAZ HERNÁNDEZ

Ciudad Nezahualcóyotl, Estado de México Febrero, 2019

A mis padres, por el apoyo incondicional que me han brindado, sin su esfuerzo y dedicación esto no sería posible. Este logro también es de ustedes.
A mi hermano por enseñarme a sonreír y a disfrutar la vida aún en la adversidad.
A mis profesores, por compartir conmigo todo su conocimiento y por su infinita paciencia.
A Dios.

De corazón, gracias.

Contenido

I. Introducción.

La creación de Internet significo una revolución en todos los ámbitos de la vida, gracias a ella las personas tuvieron la posibilidad de acceder a información que en otras épocas resultaba demasiado complicada de obtener, además de brindarle a los individuos una vía de comunicación nunca antes vista que derrumbo las barreras del tiempo y la distancia. Lo anterior trajo como consecuencia que la información se transmitiera más rápido y por ende el avance tecnológico se aceleró.

En la actualidad Internet sigue creciendo, ofreciendo servicios que no sólo se limitan a la publicación de información, sino que además dan la posibilidad de almacenarla para que esté disponible para nosotros desde cualquier dispositivo que cuente con una conexión a Internet. Por otro lado, la comunicación también se ha desarrollado pues ahora existe algo llamado *Internet of Things* (IoT) o en español *Internet de las cosas* (IdC). El concepto de IdC tiene sus raíces en 1999, fue propuesto en el Auto-ID Center del MIT (Evans, 2011) (Ashton, 2009) y se refiere a la conexión de objetos cotidianos con una red de Internet. Dicha de otra manera, con IdC podremos saber en todo momento en dónde se encuentra cada objeto que esté conectado a la red, cuáles de ellos se encuentran encendidos y cuáles están funcionando correctamente, también permitirá controlar sus funciones en todo momento o analizar la información recogida por los mismos, esto sin duda tiene una cantidad de aplicaciones prácticas bastante amplia que van desde seguridad en inmuebles, monitoreo, lucha contra el terrorismo, exploración y rescate, etc.

En el presente trabajo se pretende estudiar el comportamiento de un sistema robótico teleoperado, especialmente enfocándonos en los tiempos de retraso que se generan al utilizar en redes LAN y WAN las tecnologías Ethernet y Wi-Fi disponibles actualmente.

1.1 Objetivo.

Estudiar el comportamiento de un sistema robótico teleoperado utilizando los protocolos Ethernet y Wi-Fi en redes LAN y WAN.

1.2 Objetivos específicos.

- Estudiar el tiempo de retraso al enviar señales desde una ubicación local a una remota para mover un brazo robótico.
- Analizar la influencia que algunos buscadores de Internet tienen en el tiempo de carga y respuesta del robot.
- Establecer la importancia de los proveedores de Internet al montar un servidor que permita acceso desde el exterior de la red.
- Marcar pautas para la adecuada selección del dispositivo que funcionará como servidor, pensando en una adecuada eficiencia y facilidad de uso.
- Aportar a la FES un modelo de enseñanza.

II. Antecedentes.

2.1 Robots industriales.

2.1.1 Definición.

Cuando hablamos de *robots industriales* suele venir a nuestra mente enormes brazos mecánicos capaces de levantar varios kilos, así como de realizar el ensamblaje o manufactura de productos, sin embargo, el término *robot industrial* más bien se refiere a los llamados *robots manipuladores* que si bien pueden ser de tamaños bastante grandes también pueden tener dimensiones menores que se ajusten al tipo de tarea a realizar.

Debido a que el presente trabajo se enfoca en el estudio de un sistema que tiene como actuador un pequeño robot manipulador, es necesario ofrecer al lector una definición que sirva como introducción al análisis de dichos sistemas. Dentro del mundo de la robótica suelen darse diversas definiciones para el concepto de *robot industrial*, dichas definiciones están influenciadas en su mayoría por la geográfica en la cual se realizan debido a que el mercado japonés y el mercado euro-americano presentan diferencias en cuanto a lo que cada uno considera un robot industrial; para los japoneses un robot industrial es cualquier mecanismo que posea articulaciones móviles y que es utilizado para la manipulación, mientras que para el mercado euro-americano es necesario cierto nivel de complejidad especialmente en el control del dispositivo.

La Robotic Industries Association (RIA, 1979) define robot industrial como:

> *"Un manipulador reprogramable y multifuncional diseñado para mover material, piezas, herramientas o dispositivos especializados a través de diversas trayectorias programadas para la realización de una variedad de tareas."*

A pesar de que dicha definición excluye otros tipos de robótica, se ajusta a nuestras necesidades pues se centra en la llamada *robótica de manipulación.*

2.1.2 Partes.

Un robot manipulador se encuentra conformado por 2 elementos en los cuales se puede englobar toda su estructura. Dichos elementos son:

1. Manipulador (también llamado "brazo mecánico")
2. Controlador.

A su vez, cada uno de ellos está compuesto una variedad de partes que funcionan en conjunto para asegurar el correcto funcionamiento del sistema.

Manipulador.

Es la parte mecánica del sistema, está compuesto por una serie de motores, engranajes, ejes y actuadores que permiten soportar el movimiento de las partes del manipulador, estas partes se conocen como *eslabones* y varían de un modelo a otro, aunque en general son cinco:

- Base: Es la parte a la que se encuentra fijo el robot, usualmente tiene un movimiento rotatorio.
- Cuerpo: En algunos mecanismos esta parte no tiene movimiento y sirve únicamente como unión entre el antebrazo y la base.
- Brazo.
- Antebrazo.
- Herramienta: Es la parte terminal del sistema y aunque suele ser una pinza que permite manipular objetos no siempre es así. A la parte final de dicha herramienta se le llama *elemento terminal.*

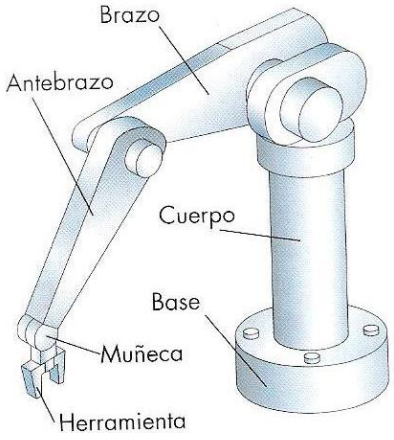


Figura 1: Estructura de un robot industrial. Recuperado de (Jaime, 2016).

Estas partes se encuentran unidas entre sí por *articulaciones*, debido a que su morfología es similar a la de un brazo humano a cada articulación se le ha nombrado relacionándola con una articulación humana, de esa manera existen por lo general tres articulaciones:

1. Hombro.
2. Codo.
3. Muñeca.

Al conjunto de eslabones y articulaciones se le denomina *cadena cinemática.* Una cadena cinemática es abierta si cada eslabón está conectado a otro cualquiera por un solo camino. Por el contrario, se le llamara cerrada si cada eslabón se encuentra unido a cualquier otro por al menos dos caminos diferentes formando uno o más lazos cerrados (Cisneros Limón, 2006).

Los actuadores encargados de brindarle movimiento a las articulaciones dependen de la aplicación que se le quiera dar al sistema, de esa manera los actuadores eléctricos se utilizan en sistemas que requieren de una gran precisión además de que son más fáciles de controlar, los actuadores neumáticos ofrecen una respuesta rápida, aunque carecen de precisión; mientras que los actuadores hidráulicos son adecuados para sistemas que requieren una gran capacidad de carga y una buena regulación de la velocidad (Universidad de Santiago de Chile, s.f).

Las articulaciones pueden ser de tipo lineal o rotacional. La llamaremos lineal si un eslabón se desliza sobre el eje solidario al eslabón anterior, también se le conoce como deslizante, traslacional o prismática.

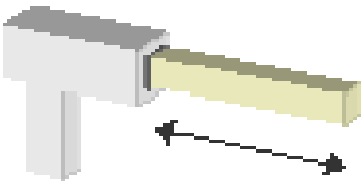


Figura 2: Ejemplo de articulación lineal. Recuperado de (Márquez Aguilera & Giron Bobadilla, 2018).

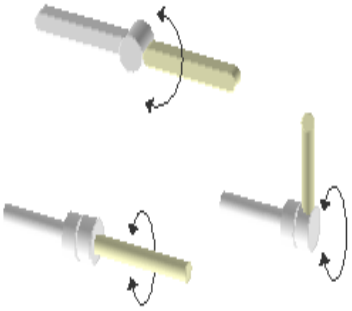


Figura 3: Ejemplos de articulaciones rotacionales. Recuperado de (Márquez Aguilera & Giron Bobadilla, 2018).

Se le denomina rotacional si el eslabón gira en torno al eje solidario del eslabón anterior (Márquez Aguilera & Giron Bobadilla, 2018).

Existe otro tipo de articulación llamada *esférica* que permite la realización de movimientos según los 3 ejes (Vigil Sanabria, 2009).

Controlador.

En este punto es posible que se generen algunas confusiones debido a que cada autor puede dar una clasificación de los elementos que se incluyen dentro del controlador, sin embargo, en nuestro caso llamaremos controlador al conjunto de elementos electrónicos que se encargan de realizar las siguientes tareas:

- Recibir y procesar la información enviada por los sensores.
- Ordenar la ejecución de rutinas.
- Ajustar el sistema basándose en la información recibida.
- Realizar la comunicación con el operador del sistema.
- Almacenar la información para su posterior estudio.

La consideración anterior se hará sin perder de vista que es común agrupar los elementos en grupos dependiendo de la función que realizan.

Es posible deducir que el número de tareas necesarias para la correcta realizar de una determinada labor depende del sistema y de la complejidad del trabajo que se desee realizar, de ese modo habrá situaciones en las que, por ejemplo, el sistema no contará con sensores y será directamente controlado por un operador humano o en las cuales se requerirá aplicar complejos métodos de control automatizado para asegurar la correcta realización del trabajo.

2.1.3 Grados de libertad.

La definición de grado de libertad (GDL) puede llegar a ser bastante abstracta, sin embargo para nosotros un GDL será “cada una de las coordenadas independientes que son necesarias para describir el estado del sistema mecánico del robot (posición y orientación en el espacio de sus elementos)” (Márquez y Giron, 2018, Grados de libertad, *Inteligencia artificial*, recuperado de https://freedoomforlife.wordpress.com/grados-de-libertad/). En otras palabras, los

grados de libertad no son más que los parámetros necesarios para determinar la posición y orientación del elemento terminal del robot manipulador.

Debido a que las articulaciones más utilizadas son prismáticas y rotacionales con un grado de libertad cada una, el número de GDL suele coincidir con el número de articulaciones.

2.1.4 Configuraciones.

A la hora de diseñar un robot industrial pueden utilizarse combinaciones de los distintos tipos de articulaciones con el objetivo de optimizar su funcionamiento y asegurar la correcta realización del trabajo, dichas combinaciones han dado lugar a una serie de configuraciones mecánicas que se relacionan con los modelos de coordenadas en el espacio las cuales son: cartesianas, cilíndricas, esféricas o polares y angulares. Con base en lo anterior existen cuatro configuraciones llamadas clásicas, que son (González, 2002):

1. Cartesiana.
2. Cilíndrica.
3. Esférica o polar.
4. Angular o *de brazo articulado.*

Y una no clásica:

- SCARA (Selective Compliance Assembly Robot Arm) (Alejandro & Miguel, 2017).

Cartesiana.

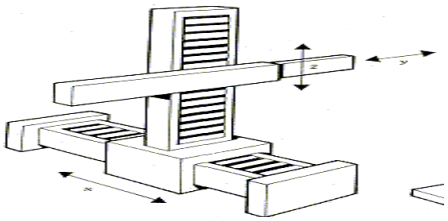


Figura 4: Configuración cartesiana. Recuperado de (Alejandro & Miguel, 2017).

Utiliza articulaciones prismáticas que le proporcionan tres grados de libertad que son movimientos lineales que corresponden a los movimientos sobre los ejes x, y, z. Funciona muy bien cuando se tiene un espacio de trabajo grande. Los movimientos de este robot son con base en *interpolaciones,* es decir, el tipo de trayectoria que realiza el manipulador para ir de un punto a otro. En este caso son interpolaciones lineales debido a que dichas trayectorias son en línea recta (Alejandro & Miguel, 2017).

Cilíndrica.

Está diseñado para tener 3 grados de libertad que se traducen en dos movimientos lineales y uno rotacional los cuales se llevan a cabo mediante interpolación lineal e interpolación por articulación, esta última se realiza atreves de la primera articulación pues puede ejecutar un movimiento rotacional (Alejandro & Miguel, 2017).

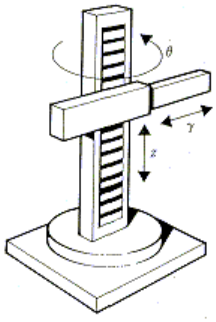


Figura 5: Configuración cilíndrica. Recuperado de (Alejandro & Miguel, 2017).

Esférica o polar.

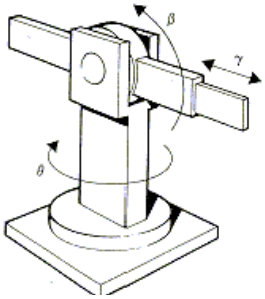


Figura 6: Configuración esférica o polar. Recuperado de (Alejandro & Miguel, 2017).

Este robot puede realizar 3 tipos de movimiento: angular, rotacional y lineal utilizando para ello la interpolación por articulaciones en sus primeras articulaciones y la interpolación lineal en la tercera (Alejandro & Miguel, 2017).

Angular o *de brazo articulado.*

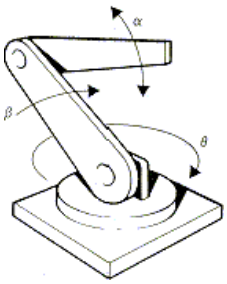


Figura 7: Configuración angular. Recuperado de (Alejandro & Miguel, 2017).

Es parecido al brazo humano, incluso sus articulaciones llevan el nombre de las articulaciones presentes en nuestros brazos (hombro, codo, muñeca). Posee generalmente 3 articulaciones, una rotacional y dos angulares que le confieren de un volumen de trabajo esférico. Sus desplazamientos son realizados mediante interpolación por articulaciones (Alejandro & Miguel, 2017).

SCARA.

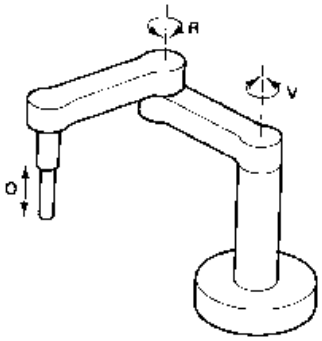


Figura 8: Configuración SCARA. Recuperado de (Alejandro & Miguel, 2017).

Es el ejemplo más común de configuración no clásica, tiene tres articulaciones, las dos primeras son rotacionales y le permiten realizan movimientos horizontales de mayor alcance, mientras que la tercera le permite realizar un movimiento lineal.

2.2 Sistema robótico teleoperado.

2.2.1 Concepto.

A estos sistemas también se les conoce como *telerobótica* y son el resultado de años de investigación y desarrollo que permitieron combinar la telepresencia con el telecontrol

Se componen principalmente de 4 elementos: operador, maestro, esclavo y medio de transmisión.

El operador u operadores son aquellas personas que se encargan de manipular el sistema desde una estación de trabajo.

Llamaremos maestro al dispositivo que funciona para transmitir las ordenes que el operador le da al sistema, el maestro puede ser un dispositivo físico (un robot, un control remoto, etc.), un sistema en línea (página web, aplicaciones para dispositivos móviles) o una combinación de estos dos.

Mientras tanto, el esclavo es el dispositivo que ejecuta las órdenes dadas por el operador y se encuentra a cierta distancia del mismo, dicho esclavo es generalmente un robot manipulador, aunque también puede ser un vehículo o un dron.

El medio de transmisión, también llamado sistema de transporte se define como la vía por la cual se transmite la información desde la estación de trabajo hasta el esclavo y viceversa.

Entonces podemos realizar la siguiente definición:

"Un sistema robótico teleoperado es aquel que permite al operador controlar de manera remota un esclavo a través de un maestro."

En el diagrama de la figura 9 se ilustra el funcionamiento de los sistemas teleoperados:

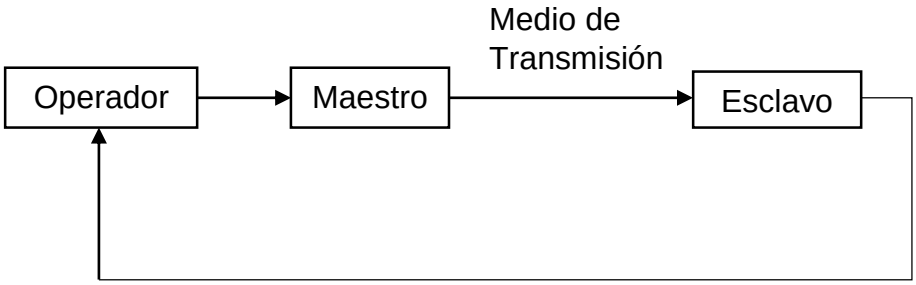


Figura 9: Diagrama de funcionamiento de un Sistema Teleoperado. Elaboración propia.

Pueden ser utilizados en situaciones en las cuales es necesario mantener al operador protegido de ambientes hostiles o cuando por cuestión de dimensiones o distancias resulta imposible que el operador se encuentre de manera física en dicho lugar, con lo anterior queda claro que la distancia que separa al operador del esclavo varía dependiendo de la aplicación que se desea darle al sistema.

Figura 10: Sonda espacial Curiosity. Recuperado de (El Universal, 2013).

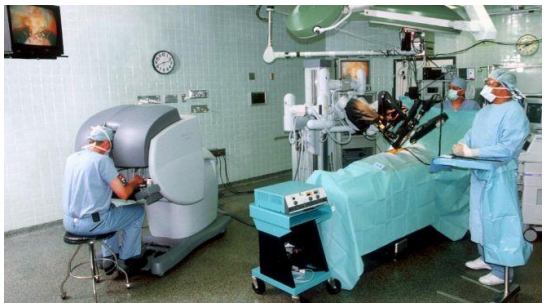


Figura 11: Robot Da Vinci. Recuperado de (Morente, 2017).

2.2.2 Telepresencia.

El término de telepresencia puede significar diferentes cosas dependiendo del contexto en el que es utilizado, sin embargo, en general se define como la sensación de estar físicamente en un lugar lejano (Noticias Multimedia, 2015).

En un mundo cada vez más globalizado, la telepresencia se presenta como la evolución obvia y directa de las videoconferencias resultado de la necesidad de mantener en contacto a un grupo de personas que se encuentran trabajando en un mismo proyecto y al mismo tiempo están ubicados en diferentes sedes geográficas, lo anterior suponía un nuevo reto debido a que la comunicación en las videoconferencias es bidireccional y se complica a la hora de abarcar un número mayor de sedes, por tal motivo se inició el desarrollo de la telepresencia, la cual permite una comunicación multicanal (TicWeb, 2015).

A pesar de que ambos términos (teleconferencia y telepresencia) suelen usarse por igual, la diferencia entre ambos radica en que la telepresencia requiere de un amplio ancho de banda, pantalla con resolución Full HD o UHD 4K, una habitación con varias cámaras, micrófonos e iluminación adecuada (Noticias Multimedia, 2017) que proporcionen la sensación de “estar ahí”.

Algunas de sus ventajas son (Techno Trends, s.f.):

- Reducción de gastos.
- Ahorro de tiempo.
- Aceleración de la toma de decisiones.
- Mayor y mejor comunicación.

En la actualidad se han desarrollado un tipo de robots llamados *robots de telepresencia* que están compuestos generalmente por una o varias pantallas, cámaras y micrófonos montados sobre una base móvil que es controlada por el usuario y cuyo objetivo es romper con las limitaciones actuales de esta tecnología.

2.2.3 Telecontrol

Dicha palabra proviene de la raíz griega *tele* que significa *distancia* y básicamente se refiere al control de sistemas o aparatos, ejercido a distancia (Real Academia Española, 2017).

El telecontrol se diferencia de la telemetría en el sentido de que esta se enfoca únicamente en tomar la información de parámetros de funcionamiento de una red y centralizarla para su conocimiento, almacenamiento y uso posterior (Glosario de Riego, 2018) mientras que el telecontrol permite la manipulación de los actuadores mediante ordenes que pueden ser dadas por un operador o bien por un sistema que le brinde autonomía a la red.

2.3 Redes de datos.

2.3.1 El modelo de referencia OSI.

En 1978 la International Organization for Standardization (ISO) publicó por primera vez el modelo de referencia para la Interconexión de Sistemas Abiertos (Open System Interconnection, OSI). Su objetivo principal fue impulsar la *interoperabilidad*, es decir, la posibilidad de que sistemas incompatibles puedan trabajar juntos, por ejemplo, una LAN Ethernet intercambiando mensajes con una LAN Token Ring. También fue la guía utilizada para diseñar el Internet Protocol (Protocolo de Internet, IP) y actualmente es el estándar utilizado para diseñar métodos de comunicación entres dispositivos de red (Velte & Velte, 2008).

Se estructura en niveles o capas en los cuales se descompone el proceso de comunicación para hacerlo más sencillo. Cada capa se encarga de resolver problemas específicos sin preocuparse por tratar con problemas pertenecientes a otros niveles, está organizada de manera jerárquica, la capa inferior se comunica con la capa superior inmediata y viceversa sin tener la posibilidad de saltarse niveles (Barbancho, y otro, Redes locales (segunda ed.), 2014).

Las capas son:

1. Física.
2. De vinculación de datos.
3. De red.
4. De transporte.
5. De sesión.
6. De presentación.
7. De la aplicación.

Aplicación.
Presentación.
Sesión.
Transporte
Red.
Vinculación de datos.
Física.

Tabla 1: Organización jerárquica de la pila OSI. Elaboración propia.

Capa física.

Se encarga de las especificaciones mecánicas, eléctricas, funcionales y de procedimientos de la transmisión física (Barbancho, y otro, Redes locales (segunda ed.), 2014).

Capa de vinculación de datos.

Su trabajo es asegurar que los mensajes entres dos puntos de la red lleguen sin errores, sin importar la tecnología de transmisión física empleada (Barbancho, y otro, Redes locales (segunda ed.), 2014).

Capa de red.

En esta capa se encuentran las funciones que establecen el camino real por el cual viajarán los datos (Barbancho, y otro, Redes locales (segunda ed.), 2014).

Capa de transporte.

En esta capa se encuentran representadas las funciones que proporcionan mecanismos de seguridad, recuperación de errores y control de flujo entre puntos finales (Barbancho, y otro, Redes locales (segunda ed.), 2014).

Capa de sesión.

Funciones que establecen la conversación, los turnos de palabra, asentamientos, control del intercambio de datos, etc. Para controlar la comunicación entre aplicaciones (Barbancho, y otro, Redes locales (segunda ed.), 2014).

Capa de presentación.

Aquí se representa a las aplicaciones encargadas de traducir en diferentes representaciones la información empleada por las aplicaciones (Barbancho, y otro, Redes locales (segunda ed.), 2014).

Capa de aplicación.

Funciones que proporcionan acceso al entorno OSI (Barbancho, y otro, Redes locales (segunda ed.), 2014).

2.3.2 Protocolos TCP/IP

Fueron desarrollados en 1973 por Vinton G. Cerf y Robert Kahn para la DARPA (Defense Advance Research Projects Agency), sin embargo, no fueron publicados hasta mayo de 1974 en la revista IEEE Transactions on Communications

Technology (Martínez, 2012). En realidad, bajo esas siglas se agrupa un paquete de protocolos de comunicación de datos, dicho paquete toma su nombre del Transmission Control Protocol (Protocolo de Control de Transmisión) e Internet Protocol (Protocolo de Internet), dos de los más importantes protocolos que podemos encontrar incluidos en él (Textos científicos, 2006).

Los protocolos TCP/IP presentan las siguientes características:

- Son estándares de protocolos abiertos y gratuitos cuyo desarrollo y modificaciones se realizan por consenso.
- Independencia a nivel de software y hardware, lo que los hace idóneos para interconectar equipos de diferentes fabricantes, no sólo a Internet sino también formando redes locales.
- Proporcionan un esquema común de direccionamiento que permite a un dispositivo con TCP/IP localizar a otro en cualquier punto de la red.
- Son protocolos estandarizados de alto nivel (científicos, 2006).

A pesar de lo anterior, TCP/IP no cubre todos los estándares que existen en comunicaciones.

Separando el TCP/IP en capas es posible realizar una comparativa con el modelo OSI, como se muestra a continuación:

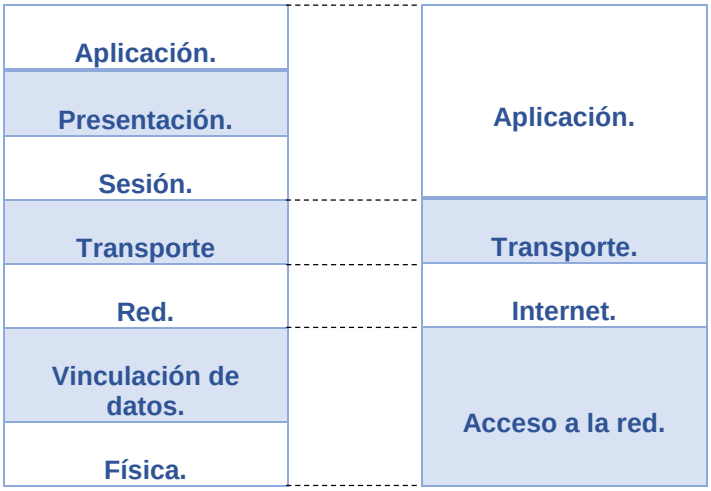


Tabla 2: Comparación entre el modelo OSI y el TCP/IP. Elaboración propia.

Capa de Acceso a la red.

Incluye las funciones de la capa física y la capa de vinculación de datos del modelo OSI. Trata sobre las tecnologías y protocolos WAN y LAN, como Frame Relay y Ethernet, requeridos para realizar el enlace físico (Valencia & Calixto).

Capa de Internet.

También es conocida como capa de red o capa IP, acepta y transfiere paquetes para la red. Incluye el Protocolo de Internet, el ARP (Address Resolution Protocol, Protocolo de Resolución de Direcciones) y el ICMP (Internet Control Message Protocol, Protocolo de Mensajes de Control de Internet) (Oracle, 2010).

El IP es considerado un *protocolo sin conexión* debido a que no intercambia información de control para establecer una conexión antes de enviar los datos, en caso de que se necesitara de dicha conexión, éste delegará esa labor a protocolos de otras capas (Textos científicos, 2016).

Este protocolo tampoco realiza detección de errores o recuperación de datos ante los mismos (Textos científicos, 2016).

Las funciones que realiza el IP incluyen:

- Definir el datagrama, el cual es el formato de los paquetes de información enviados y utilizados por el Protocolo de Internet.
- Definir el esquema de direccionamiento de Internet.
- Mover los datos entre las capas adyacentes.
- Ruteo: encausar los datagramas hacia sistemas remotos.
- Realizar fragmentación y re-ensamblaje de los datagramas (Textos científicos, 2016).

<table>
<tr><td>Bits 0 - 3</td><td>4 - 7</td><td>8 - 15</td><td>16 - 18</td><td>19 - 31</td></tr>
<tr><td>Versión</td><td>Longitud Encabezado IP</td><td>Tipo de servicio</td><td colspan="2">Longitud Total</td></tr>
<tr><td colspan="3">Identificación</td><td>Flags</td><td>Offset del fragmento</td></tr>
<tr><td colspan="2">Tiempo de vida</td><td>Protocolo</td><td colspan="2">Chequeo de cabecera</td></tr>
<tr><td colspan="5">Dirección de origen</td></tr>
<tr><td colspan="5">Dirección de destino</td></tr>
<tr><td colspan="5">Opciones</td></tr>
<tr><td colspan="5">Datos</td></tr>
</table>

Tabla 3: Estructura de un paquete IPv4. Recuperado de (Textos científicos, 2016).

El ARP ayuda al IP a dirigir los datagramas al sistema receptor asignando direcciones Ethernet a direcciones IP conocidas (Oracle, 2010).

Mientras tanto el ICMP detecta y registra las condiciones de error de la red, algunas de las cosas que registra son:

- Paquetes soltados.
- Fallos de conectividad.
- Redirección (Oracle, 2010).

Así mismo, los mensajes enviados por este protocolo realizan las siguientes funciones:

- Control de flujo.
- Detección de destinos inalcanzables.
- Redirección de rutas.
- Chequeo de sistemas remotos (Textos científicos, 2016).

Capa de Transporte.

Esta capa asegura que los paquetes lleguen en secuencia y sin errores, al intercambiar la confirmación de la recepción de los datos y retransmitir los paquetes perdidos (Oracle, 2010).

Los principales protocolos que se encuentran en esta capa son el TCP, SCTP (Stream Control Transmission Protocol, Protocolo de Transmisión para el Control de Flujos) y el UDP (User Datagram Protocol, Protocolo de Datagramas de Usuario).

El TCP es un protocolo *orientado a la conexión* que permite a las aplicaciones comunicarse entre sí y ve los datos que envía como un flujo continuo de bytes, no como paquetes independientes, motivo por el cual es necesario enviarlos en la secuencia correcta (Textos científicos, 2016).

SCTP ofrece los mismos servicios que TCP y además permite conexiones entres sistemas que permiten host múltiple (más de una dirección) (Oracle, 2010).

El protocolo UDP es poco confiable pues no verifica las conexiones entre receptor y transmisor, sin embargo, resulta ideal para las aplicaciones que envíen pequeñas cantidades de datos (Oracle, 2010).

Capa de Aplicación.

Esta capa abarca las capas de aplicación, presentación y sesión del modelo OSI. Aquí se incluyen los procesos que usan los protocolos de la capa de transporte. La mayor parte proporcionan servicios de usuarios. Algunos ejemplos son:

- Telnet.
- FTP.

- SMTP.

Los cuales hacen uso de los servicios orientados a la conexión del TCP.

Algunos protocolos que usan los servicios del UDP son:

- DNS.
- NFS.
- RIP (Textos científicos, 2016).

2.3.3 Tipos de redes.

Las redes pueden ser clasificadas según distintos criterios, sin embargo, en este caso se clasificarán según su área de distribución, es decir, según su tamaño:

- BAN.
- PAN.
- LAN.
- MAN.
- WAN

Body Area Network (BAN).

Estas redes son parte de las PAN y se pueden encontrar dentro, cerca o alrededor de una persona o ser vivo. En ellas es posible encontrar algunas tecnologías como la Bluetooth o la Ultra Wide Band. Tienen una cobertura no superior a dos o tres metros y emiten una muy baja cantidad de energía, lo cual contribuye a una larga vida de la batería de los dispositivos, la reducción de los niveles de interferencia y la operación con potencias que no representan un riesgo para los seres vivos.

Sus principales aplicaciones se encuentran en la medicina y el entretenimiento. En el primer caso permiten el monitoreo de las variables del cuerpo del ser vivo para diagnosticar enfermedades, gestionar y controlar tratamientos, mejorar la interacción de personas con limitaciones físicas, etc. En el segundo caso el principal ejemplo se encuentra en la transferencia de información multimedia entre dispositivos inteligentes (Betancur, 2011).

Personal Area Network (PAN).

Al igual que las BAN, se emplean dentro de lo que se ha denominado *espacio operativo personal,* es decir, el espacio que rodea a una persona. Cubren distancias de pocos metros. Generalmente se usan para conectar dispositivos periféricos como impresoras, teléfonos móviles y electrodomésticos o un asistente personal digital (Personal Digital Assistant, PDA) a un ordenador (Barrenechea, 2011). Las

tecnologías utilizadas son básicamente las mismas que en las BAN y sus aplicaciones son bastante similares.

Es necesario mencionar que, si bien los términos BAN y PAN son correctos, en la práctica lo más común es utilizar redes WBAN (Wireless Body Area Network) y WPAN (Wireless Personal Area Network) debido a las ventajas que las redes Wireless ofrecen.

Local Area Network (LAN).

Según la IEEE (Institute of Electrical and Electronics Engineers), una LAN se define como un sistema de comunicación de datos que permite a un número de dispositivos independientes comunicarse directamente con cualquier otro, dentro de un área geográfica de tamaño moderado y sobre un canal de comunicaciones físico con un rango moderado de información (Martín, Kavanagh, & Leben, Local Area Networks, 1994).

De la anterior definición es posible resaltar algunos puntos importantes. En primer lugar, las LAN usualmente soportan un tipo de comunicación llamada *many-to-many,* en la cual cualquier dispositivo conectado a la LAN podrá comunicarse directamente con cualquier otro que se encuentre dentro de la misma red sin la necesidad de la intervención de un dispositivo central encargado de controlar la comunicación y que es considerado como más inteligente que los dos anteriores (Martín, Kavanagh, & Leben, Local Area Networks, 1994).

Otro punto interesante es que las LAN se definan dentro de áreas de tamaño moderado, este es el rasgo que más claramente diferencia una LAN de una WAN, pues la LAN usualmente está limitada a menos de un kilómetro, conectando a dispositivos que se encuentran relativamente cerca entre sí (Martín, Kavanagh, & Leben, Local Area Networks, 1994).

Además de lo antes mencionado, otra característica de este tipo de redes es el uso de un canal de comunicaciones privado dedicado únicamente a la red, dicho canal puede ser cableado o vía inalámbrica (Martín, Kavanagh, & Leben, Local Area Networks, 1994).

Metropolitan Area Network (MAN).

Podemos ubicarla, por su área de cobertura, entre la LAN y la WAN La arquitectura utilizada en una red de área metropolitana suele ser muy similar a la usada en una red de área local, sin embargo, una MAN tiene un área de cobertura de entre 30 y 50 km aproximadamente. Estas redes pueden ser utilizadas para unir un LAN con una WAN (Martín, Kavanagh, & Leben, Local Area Networks, 1994).

Wide Area Network (WAN).

Es una red privada de telecomunicaciones que se encuentra distribuida geográficamente y pude interconectar varias LAN que se encuentran separadas por varios kilómetros de distancia entre sí. Normalmente se utiliza un enrutador para conectar la LAN y la WAN (Rouse, 2016).

Las tecnologías utilizadas pueden ser, al igual que en las anteriores, cableadas o inalámbricas. En las primeras destacan la conmutación de etiquetas multiprotocolo, T1s, Carrier Ethernet y enlaces comerciales de banda ancha a Internet. Mientras tanto, en las tecnologías inalámbricas se encuentran las redes de datos celulares, redes públicas Wi-Fi o satelitales (Rouse, 2016).

2.3.4 Medios de transmisión.

Es el medio físico por el que se propaga la señal que contiene la información, puede clasificarse basándonos en si la señal es guiada a través del medio o no, por lo cual suele hablarse de medios guiados y medios no guiados. Los medios guiados son aquellos compuestos por un material físico sólido por el cual se transporta la señal de información. Los medios no guiados son medios inalámbricos que se basan en la radiación de energía electromagnética, dicha energía es transmitida por un emisor y recibida por un receptor.

Los medios guiados que más se utilizan en telecomunicación son:

1. Par trenzado.
2. Cable coaxial.
3. Fibra óptica.

Por otro lado, los medios no guiados más comunes son:

1. Ondas de radio.
2. Microondas.
3. Infrarrojo.
4. Láser.

Par trenzado.

Está formado por 2 conductores, generalmente de cobre, cada conductor posee un aislamiento de plástico que lo recubre. Ambos hilos se trenzan de acuerdo a un número de vueltas por centímetro.

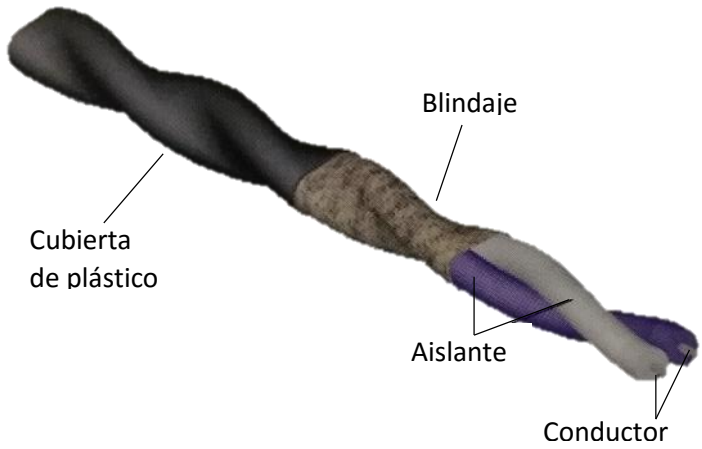


Figura 12: Par trenzado. Recuperado de (Barbancho, y otros, Medios de Transmisión, 2014).

Los principales parámetros que influyen en sus características son el grosor de los hilos, el número de vueltas del trenzado, el tipo de aislamiento, la impedancia y el material del recubrimiento (Barbancho, y otros, Medios de Transmisión, 2014).

Usualmente el número de vueltas por centímetro es 2.36.

Suele usarse en topologías en estrella, bus o anillo debido a que la velocidad de transmisión disminuye rápidamente con la distancia entre dispositivos. A pesar de esa desventaja, su bajo costo de fabricación y su fácil instalación han hecho que sea el principal medio de transmisión para el acceso telefónico a redes de voz y datos (LAN y DSL) (Barbancho, y otros, Medios de Transmisión, 2014).

Si el cable posee más de un par trenzado se denomina multipar. Existen cables de 4, 8, 100 y 300 pares, aunque en algunas ocasiones llega a haber un número mayor de pares, pero sin superar los 2200.

Básicamente hay dos tipos de par trenzado: cable UTP (Unshielded Twisted Pair, Par Trenzado sin Blindaje) y cable STP (Shielded Twisted Pair, Par Trenzado con Blindaje). El STP es el menos utilizado debido a que posee una envoltura metálica que sirve para aislarlo de interferencias y ruidos externos, pero incrementa su grosor y su costo. Dentro de STP podemos encontrar varias formas de apantallamiento, las más comunes son:

- STP: Cable de pares con lámina individual y blindaje global.
- FTP: Cable con pantalla general laminada (sin pantalla individual).
- SFTP: Cable con doble pantalla general (trenzada y laminada), sin pantalla individual.
- F/STP: Cable con pantalla laminar general y blindaje individual (Barbancho, y otros, Medios de Transmisión, 2014).

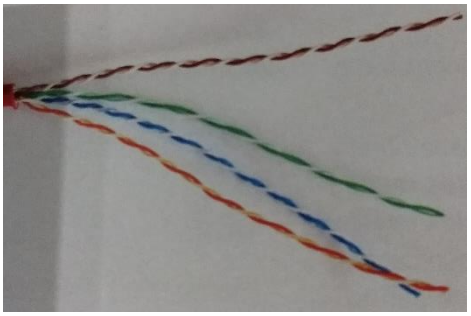


Figura 13: Par trenzado UTP. Recuperado de (Barbancho, y otros, Medios de Transmisión, 2014).

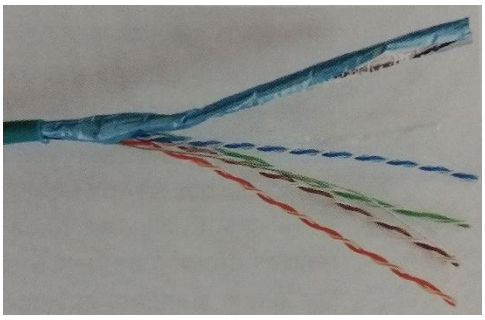


Figura 14: Par trenzado FTP. Recuperado de (Barbancho, y otros, Medios de Transmisión, 2014).

Los conectores utilizados para UTP son los denominados RJ. El RJ45 es utilizado para redes de área local, mientras que el RJ11 se usa para líneas telefónicas.

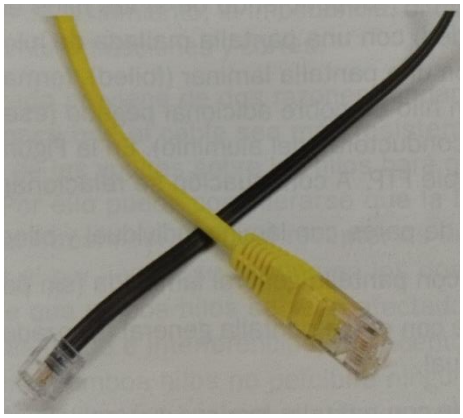


Figura 15: Conector RJ45 (amarillo) y RJ11 (negro). Recuperado de (Barbancho, y otros, Medios de Transmisión, 2014).

Cable coaxial.

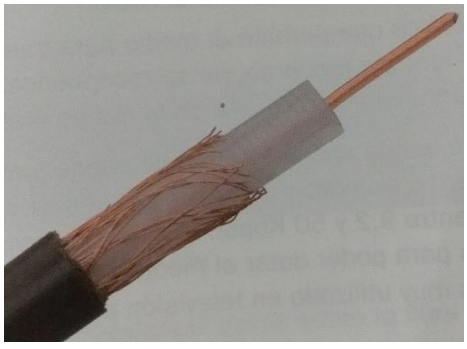


Figura 16: Cable coaxial. Recuperado de (Barbancho, y otros, Medios de Transmisión, 2014).

Está constituido por dos hilos conductores, uno de ellos es sólido y se encuentra en el centro del cable. Ese hilo se recubre de un material dieléctrico que se rodea a su vez por una hoja exterior de metal semiconductor, malla o una combinación de ambas. Este recubrimiento sirve como blindaje frente a ruido y como referencia de tensión del conductor que forma el núcleo del cable. Externamente el cable se termina con un recubrimiento plástico que sirve de aislamiento frente a contactos eléctricos e influencia de la humedad, además de proporcionarle fortaleza al cable.

Este tipo de cable es capaz de transportar señales con rangos de frecuencia más altos que los cables de pares trenzados. La atenuación que sufre la señal por la distancia es menor, además de que la protección contra interferencias externas es mayor.

Algunas de los parámetros de construcción que influyen en sus características son el grosor de los hilos conductores, la impedancia, el material de recubrimiento y la protección (Barbancho, y otros, Medios de Transmisión, 2014).

Existen dos tipos de cable coaxial:

- Cable coaxial de banda base: Posee un diámetro de aproximadamente 0.94 cm y constituye un único canal de comunicaciones que transporta una señal digital a una velocidad de transmisión muy alta, de aproximadamente 10 u 80 Mbps. Los bits no se modulan.
  Puesto que solo existe un canal, no es posible transmitir simultáneamente voz, video y datos sobre el cable.
  La distancia máxima que permite este cable para transmitir señales digitales es de 3 km, sin embargo, no se recomienda exceder los 500 m si la carga de transmisión es alta en la red de área local.
- Cable coaxial de banda ancha: Puede transportar entre 50 y 100 canales de televisión, o miles de canales de voz y de datos de baja velocidad (entre 9.2 y 50 Kbps). El espectro de frecuencia del cable se divide en dos para poder dotar al medio de bidireccionalidad.

Los conectores más comunes que se utilizan en cable coaxial son los llamados BNC (Bayonet Network Connector; Conector de Red a Bayoneta).

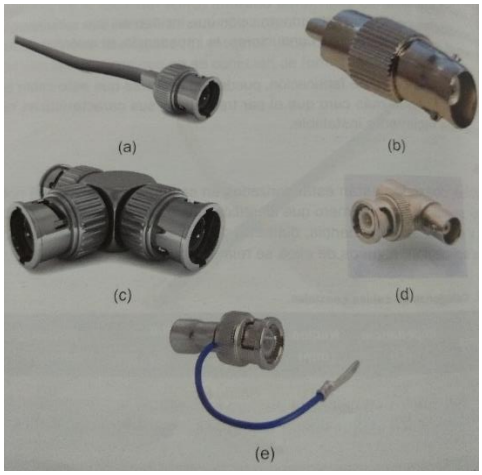


Figura 17: Conectores BNC macho (a), BNC hembra (b), BNC-T macho (c), BNC-T hembra (d) y terminador BNC (e). Recuperado de (Barbancho, y otros, Medios de Transmisión, 2014).

Fibra óptica.

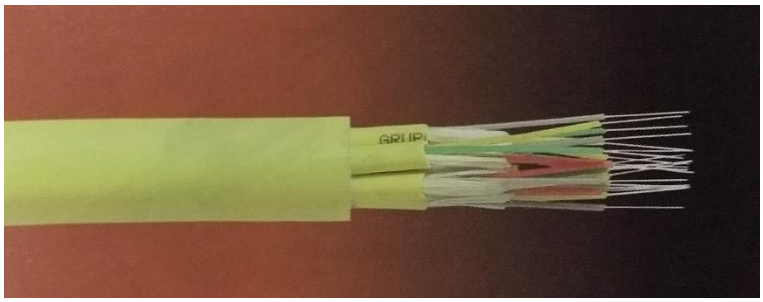


Figura 18: Fibras ópticas. Recuperado de (Barbancho, y otros, Medios de Transmisión, 2014).

Es un medio de comunicaciones constituido por plástico o cristal y su finalidad es constituir el soporte físico para el transporte de señales ópticas. Se recubre con una cubierta que protege a la fibra de roturas ante golpes y lo apantalla para evitar la influencia de otras señales luminosa. Su funcionamiento es el siguiente, la luz emitida por un transmisor incide sobre la fibra con un cierto ángulo. Si este ángulo es menor que un cierto ángulo crítico el rayo de luz se refracta y se mueve más cerca de la superficie. Si el ángulo de incidencia es mayor el rayo se refleja. Por lo tanto, es importante que el rayo se refracte para que se mueva por la fibra hasta alcanzar al receptor.

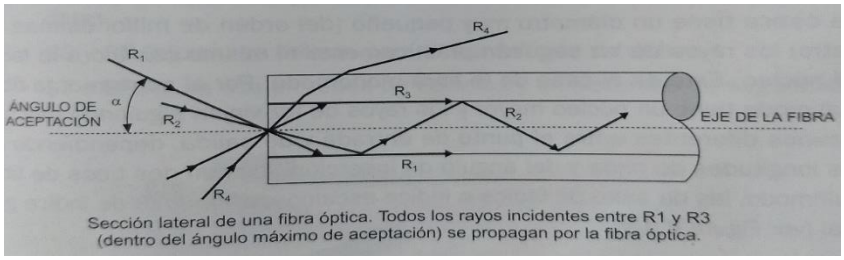


Figura 19: Funcionamiento de la fibra óptica. Recuperado de (Barbancho, y otros, Medios de Transmisión, 2014).

La fibra óptica se fabrica mediante un filamento de cristal, normalmente basado en silicio, que posee un núcleo central con un alto índice de refracción, dicho núcleo es recubierto de un material similar con un índice de refracción ligeramente menor. Esta estructura se cubre con un material aislante para evitar interferencias entre filamentos adyacentes y proporcionar protección al núcleo.

La fibra óptica tiene varias ventajas frente al cable conductor convencional, en primer lugar, la fibra no se ve afectada ante campos electromagnéticos externos. En segundo lugar, las señales que se transportan poseen un alto ancho de banda, lo cual permite velocidades de transmisión bastante elevadas, además de eso, la fibra óptica posee una baja atenuación con la distancia.

Según la forma en la que viaja la luz por la fibra existen dos tipos básicos de cable de fibra óptica: monomodo y multimodo.

- Monomodo: Tiene un diámetro del orden de las millonésimas de metro, debido a eso los rayos de luz siguen prácticamente el mismo camino a lo largo del núcleo.

- Multimodo: El diámetro de su núcleo es mayor, lo provoca que los rayos de luz viajen siguiendo muchos caminos diferentes a lo largo del punto de entrada y salida, dependiendo de sus longitudes de onda y su ángulo de inserción. Existen dos tipos de fibra multimodo, salto de índice o índice escalonado y fibra de índice gradual.

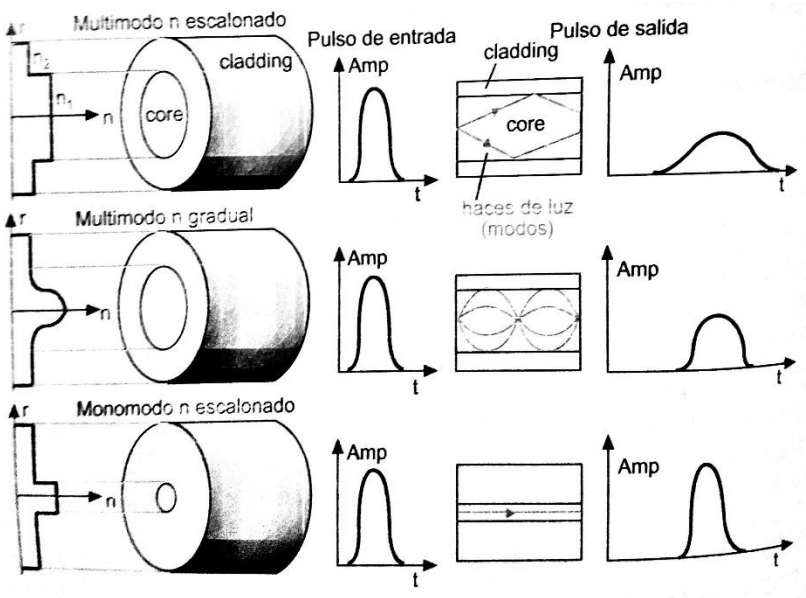


Figura 20: Fibra monomodo y fibra multimodo. Recuperado de (Barbancho, y otros, Medios de Transmisión, 2014).

Los conectores utilizados para fibra óptica son muy variados, algunos ejemplos son:

- FC: Se usa en la transmisión de datos y en las telecomunicaciones.
- FDDI: Se usa para redes de fibra óptica.
- LC y MT-Array: Utilizados en trasmisiones de alta densidad de datos.
- SC y SC-Dúplex: Es usado en la transmisión de datos.
- ST o BFOC: Se usa en redes de edificios y en sistemas de seguridad.

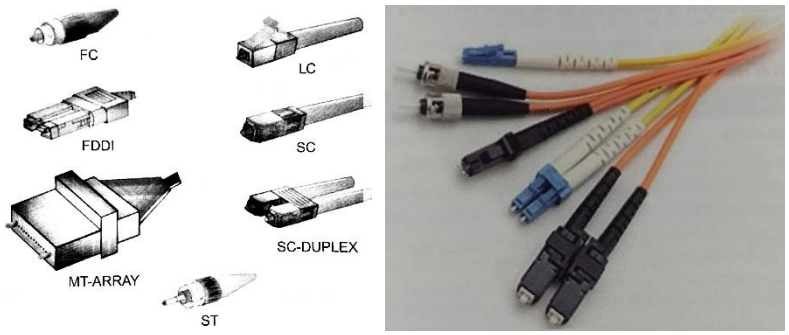


Figura 21: Conectores de fibra óptica. Recuperado de (Barbancho, y otros, Medios de transmisión, 2014).

Radio.

La mayoría de redes de ondas de radio utilizan la banda del espectro expandido comprendida entre los 10 kHz y 1 GHz, la cual esta compartida con aplicaciones como telefonía, TV, radiodifusión, etc.

Sus principales características son:

- Movilidad.
- Facilidad de conexión a la red.
- Facilidad de ampliación.
- Facilidad de integración con redes de cable.

Las ondas de radio tienen la característica de atravesar las paredes, lo cual puede ser una ventaja o una desventaja dependiendo de la aplicación que se desee darle pues puede llegar a ser complicado limitar el alcance de la red. Por tal motivo es necesario de una planificación detalla que permita su implementación con otras aplicaciones inalámbricas (Barbancho, y otros, Medios de transmisión, 2014).

Microondas.

Las microondas son unidireccionales, por lo tanto, el transmisor y el receptor deben alinearse de forma muy precisa. Las principales características de las microondas son:

- Propagación por línea de vista. Las torres con las antenas deben verse entre sí. Si la distancia entre las torres es muy grande, las torres deben ser muy altas, debido a la curvatura de la Tierra (Barbancho, y otros, Medios de transmisión, 2014).
- Las microondas de frecuencia muy alta no atraviesan las paredes.

Infrarrojo.

Las ondas infrarrojas se encuentran en la banda comprendida entre los 200 GHz y los 400 THz. Son utilizadas para comunicaciones de corto alcance debido a que no pueden atravesar paredes. Esta característica se utiliza para evitar interferencias con otras tecnologías inalámbricas. Su utilización es en interiores debido a que el sol emite radiaciones en toda la banda de infrarrojos, interfiriendo así estas comunicaciones (Barbancho, y otros, Medios de transmisión, 2014).

Láser.

Posee características similares a las del infrarrojo: uso direccional con emisor y receptor perfectamente alineados, no pueden atravesar paredes, etc. Sin embargo, puede utilizarse en exteriores, lo cual hace común su uso para conectar redes de

dos edificios que tengan visión directa entre sí. En los inconvenientes cabe destacar el elevado coste de los equipos y la gran influencia que tienen los fenómenos meteorológicos sobre las comunicaciones de este tipo (Barbancho, y otros, Medios de transmisión, 2014).

2.3.5 Ethernet.

La IEEE define Ethernet como el estándar 802.3, el cual es un protocolo de la capa de acceso a la red del modelo TCP/IP, se conoce comúnmente como protocolo CSMA/CD (Carrier Sense Multiple Access/Collision Detection, Acceso Multiple con Escucha de Portadora y Detección de Colisión). Dispone de una topología lógica de bus, es decir, la red puede estar físicamente en bus o en estrella, pero su configuración a nivel funcional es la de un medio físico compartido por todos los terminales (Huidobro, El método CSMA/CD, 2004).

Su funcionamiento es el siguiente: un ordenador antes de transmitir analiza el medio de transmisión compartido por todos los terminales conectados para comprobar si ya existe una comunicación, sino detecta ninguna comunicación, comienza la transferencia y en caso contrario, esperará un tiempo aleatorio antes de comenzar de nuevo el proceso (Huidobro, El método CSMA/CD, 2004).

Si dos o más ordenadores transmiten al mismo tiempo ocurre una *colisión*, es decir, las señales se interfieren mutuamente quedando inservibles para su correcta recepción por sus respectivos destinatarios. Al percibir dicha señal inservible, los terminales implicados en la colisión cancelan la transmisión en curso y trasmiten una secuencia especial de bits, denominada *señal de atasco*, que tiene como misión que la colisión sea detectada por el resto de terminales de la red (Huidobro, El método CSMA/CD, 2004).

Cada vez que ocurre una colisión, los terminales comienzan su proceso de transmisión después de un periodo aleatorio de espera para reducir la probabilidad de una nueva (Huidobro, El método CSMA/CD, 2004).

Modelo arquitectónico del CSMA/CD.

Para ayudar a entender la definición y funcionamiento del protocolo CSMA/CD nos apoyaremos en el modelo arquitectónico mostrado a continuación:

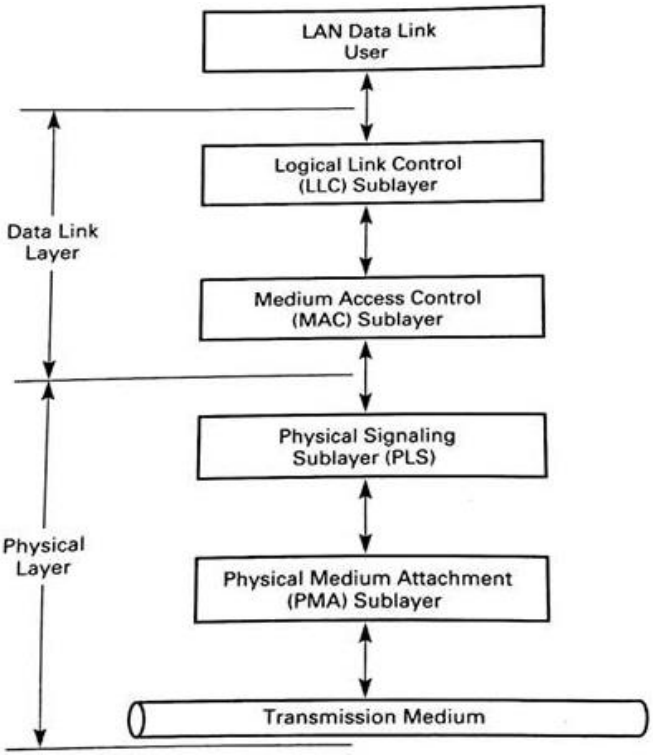


Figura 22: Modelo arquitectónico del protocolo CSMA/CD. Recuperado de (Martin, Kavanagh, & Leben, CSMA/CD Architectural Model, 1994).

Es posible observar que la capa de enlace de datos (Data Link Layer) está dividida en las subcapas de control de enlace lógico (Logical Link Control, LLC) y control de acceso al medio (Medium Access Control, MAC).

La subcapa MAC provee servicios a los usuarios que usualmente son entidades de la subcapa LLC (Martin, Kavanagh, & Leben, CSMA/CD Architectural Model, 1994).

Por su parte la capa física (Physical Layer) se divide en las subcapas de señalización física (Physical Signaling Sublayer, PLS) y la de apego al medio físico (Physical Medium Attachment, PMA) (Martin, Kavanagh, & Leben, CSMA/CD Architectural Model, 1994).

La subcapa PLS se encarga de codificar la información que es enviada desde la subcapa MAC en la estación de transmisión. Dicha acción incluye traducir los bits en las señales eléctricas correctas para ser interpretadas por el medio de transmisión. En la estación de destino, la PLS decodifica las señales recibidas y

traduce las señales eléctricas en bits para enviarlos a la subcapa MAC (Martin, Kavanagh, & Leben, CSMA/CD Architectural Model, 1994).

La PMA ofrece servicios la PLS para mejorar la función de traducción entre la PLS y el medio de transmisión, así mismo define las características de dicho medio de transmisión (Martin, Kavanagh, & Leben, CSMA/CD Architectural Model, 1994).

Velocidades actuales de Ethernet.

Actualmente se definen cuatro velocidades para la operación sobre fibra óptica y par trenzado (Huidobro, El método CSMA/CD, 2004) (Cisco, s.f.).

- Ethernet 10BaseT (10Mbps): Sobre cables trenzados sin apantallar (UTP), con topología física en estrella. El estándar especifica una longitud máxima de 90 m. para cada tramo, más 10 metros para interconexión en la estación de trabajo y en el concentrador. En 10BaseT, la transmisión de una estación de trabajo va primero al concentrador, que después la retransmite a todas las demás estaciones, además las estaciones pueden conectarse o desconectarse de los concentradores sin que el resto de las estaciones de la red resulten afectadas (Rábago, 2008).
- Fast Ethernet (100Mbps): Existen 2 estándares para esta velocidad: 100BaseFX y 100BaseSX. Ambos permiten velocidades de 100Mbps usando fibra multimodo, sin embargo, el estándar 100BaseFX usa una longitud de onda de 1300nm y permite segmentos de 2000m. Por otra parte, el estándar 100BaseSX usa una longitud de onda de 850nm lo cual permite usar un transceptor que produzca ambas longitudes de onda, 1300 y 850nm. La distancia máxima que permite 100BaseSX para cada segmento es de 500m (Rábago, 2008).
- Gigabit Ethernet (1000Mbps): Al igual que Fast Ethernet, Gigabit Ethernet utiliza dos estándares: 1000BaseSX y 1000BaseLX. Ambos permiten 100m de distancia en instalaciones horizontales y 220 a 550m en instalaciones backbone. El estándar SX utiliza longitudes de onda de 8560nm con fibra multimodo, mientras que el LX utiliza longitud de onda de 1300nm en fibra multimodo y 1310nm en fibra monomodo. Debido a que ambos utilizan fibra óptica se asegura la compatibilidad con las dos velocidades anteriores (Rábago, 2008).
- Ethernet de 10 Gigabit (10000 Mbps): Se utilizan 3 estándares: 10GbaseSR, 10GbaseLR y 10GbaseER. El SR utiliza fibra multimodo y permite distancias de hasta 300m. Mientras tanto, los protocolos LR y ER utilizan fibra monomodo y permiten distancias de 10 y 40 km respectivamente.

2.3.6 Wi-Fi.

Como ya se explicó anteriormente, las redes locales tradicionales suelen ser redes cableadas que siguen el estándar 802.3 y el protocolo CSMA/CD, sin embargo, éstas presentan una serie de limitaciones debidas al propio medio físico de enlace; fibra óptica, UTP, STP; y a los puntos de acceso fijos ubicados en lugares determinados.

Para resolver estos problemas se desarrollaron las redes WLAN (Wireless Local Area Network), sin embargo, dichas redes no seguían ningún estándar o protocolo, motivo por el cual causaban problemas de incompatibilidad entre ellas.

No fue sino hasta 1999, cuando Nokia y Symbol Technologies crearon la asociación Wireless Ethernet Compatibility Alliance (WECA, renombrada en 2003 como *WI-FI Alliance*), que se comenzó a trabajar en la creación de una marca que permitiese fomentar más fácilmente la tecnología inalámbrica y asegurar la compatibilidad de los equipos. Finalmente, un año después la WECA certifica bajo el estándar 802.11 de la IEEE al Wi-Fi (Wireless Fidelity), utilizando la banda de 2.4 GHz, alcanzando una velocidad de 11Mbps y con la capacidad para abarcar una distancia de unos pocos cientos de metros (Historia de la informática, 2010).

Versiones del estándar IEEE 802.11.

Existen varias versiones de la normativa 802.11, creadas con el objetivo de solucionar problemáticas que fueron surgiendo con el paso del tiempo; por ejemplo, algunas versiones de este estándar utilizan la frecuencia de 2.4 GHz, utilizada también por la tecnología Bluetooth para la transferencia de información, para evitar cualquier problema de interferencia se optó por crear la versión 1.2 de Bluetooth y la versión 802.11a que utiliza la frecuencia de 5 GHz (Gómez, Redes inalámbricas de área local, 2008).

Algunas de dichas versiones se mencionan a continuación:

- 802.11b. Introducida en 1999, su velocidad de transmisión es de 11Mbps y trabaja en la banda de los 2.4 GHz. Es muy sensible a la interferencia con otras tecnologías inalámbricas (Gómez, Normativa IEEE 802. 11, 2008).
- 802.11a. Trabaja en la banda de 5GHz y utiliza la técnica de transmisión conocida como Orthogonal Frequency Division Multiplexing (OFDM). Alcanza velocidades de entre 54 y 108 Mbps (Gómez, Normativa IEEE 802. 11, 2008).
- 802.11d. Permite la interacción de forma internacional de las redes 802.11 locales. Ofrece un mecanismo para que los distintos dispositivos intercambien información en los rangos de frecuencia permitidos por el país de origen del dispositivo (Ramos, Barbero, Fernández, Daswani, & Marugán, 2015).

- 802.11e. Incorpora la calidad del servicio (QoS) en la capa de enlace de datos. Define los requisitos que se van a introducir en los diferentes tipos de paquetes, definiendo un ancho de banda establecido y un retardo de transmisión (Ramos, Barbero, Fernández, Daswani, & Marugán, 2015).
- 802.11f. Incorpora el nuevo protocolo IAPP para lograr la itinerancia de usuarios en movimiento entre distintos puntos de acceso, incluso de diferentes fabricantes. Es decir, está destinado a compatibilizar diferentes productos de fabricantes de puntos de acceso y estaciones para que sus productos puedan interactuar entre sí (Ramos, Barbero, Fernández, Daswani, & Marugán, 2015).
- 802.11g. Surgió en 2003 como una evolución de la versión 802.11b. Ofrece una velocidad de 54 Mbps en la banda de los 2.4GHz y es compatible con los equipos 802.11b (Gómez, Normativa IEEE 802. 11, 2008).
- 802.11h. Compatibiliza el estándar europeo HiperLAN 2. Regula el uso de la tecnología DFS y TPC para evitar interferencias (Ramos, Barbero, Fernández, Daswani, & Marugán, 2015).
- 802.11i. Fue aprobado en 2004. Incorpora un nivel real alto de seguridad en las redes Wi-Fi para resolver sus problemas de vulnerabilidad. Incorpora CCMP basado en el Advanced Encryption Standard (AES, Estándar de Cifrado Avanzado). Permite cifrar las transmisiones de los estándares 802.11a, 802.11b y 802.11g. Se conoce como WPA2 (Ramos, Barbero, Fernández, Daswani, & Marugán, 2015).
- 802.11Ir. Fue aprobado en 2004 y permitía el uso de señales infrarrojas. Ha quedado tecnológicamente obsoleto con el tiempo (Ramos, Barbero, Fernández, Daswani, & Marugán, 2015).
- 802.11j. Adapta 802.11 a la regulación de radiofrecuencias de Japón (Ramos, Barbero, Fernández, Daswani, & Marugán, 2015).
- 802.11n. Fue ratificado en 2009. Pretende obtener anches de banda reales de hasta 300Mbps. Se implementa la tecnología MIMO (Multiple Input – Multiple Output) que permite utilizar varios radios, varias antenas y varios canales a la vez. No se centra en un solo ancho de banda, sino que permite el uso de las bandas de 2.4 y 5 GHz (Ramos, Barbero, Fernández, Daswani, & Marugán, 2015).
- 802.11p. Ofrece un modo de trabajo adecuado para su uso en dispositivos de gran movilidad. Su frecuencia de trabajo se encuentra entre los 5.90 y 6.20 GHz (Ramos, Barbero, Fernández, Daswani, & Marugán, 2015).
- 802.11ah. Aprobado en 2016 regula el empleo de redes de uso público en frecuencias inferiores a 1 GHz (IEEE, 2018).

Es necesario que destacar que algunos de los estándares mencionados anteriormente se encuentran en desuso o han sido superados (IEEE, 2018) por otros con el objetivo de mantener el 802.11 tecnológicamente actualizado.

Arquitectura.

Las WLAN basadas en el estándar 802.11 presentan una estructura similar a las redes de telefonía móvil. Cada celda, también llamada BSS (Basic Service Set), se encuentra gobernada por una estación base o punto de acceso (AP, Access Point), conectados entre sí a través de una Red Troncal de Distribución o DS (Distribution System). En algunas ocasiones los DS se agrupan en niveles jerárquicos superiores formando un ESS (Extended Service Set). Lo anterior ha originado una serie de alternativas a la hora de configurar la estructura de las redes (Huidobro & Roldán, Principios básicos de 802.11, 2006).

Una posibilidad es una red aislada de otras o IBSS (Isolated BSS) en la cual no existe sistema de distribución, sino que el AP única hace de intermediario para la conexión de los dispositivos que se encuentran dentro de su área de cobertura redes (Huidobro & Roldán, Principios básicos de 802.11, 2006).

La variante conocida como *ad hoc* es en la cual no existe AP, en dicha variable los dispositivos móviles se comunican directamente entre sí y las funciones de coordinación son asumidas por uno de ellos redes (Huidobro & Roldán, Principios básicos de 802.11, 2006).

Para terminar con los ejemplos tenemos a la configuración más habitual para redes 802.11, la cual está basada en punto de acceso. Este tipo de redes requiere de una mayor planificación ya que deben de asegurarse aspectos como un ancho de banda mínimo para cada usuario redes (Huidobro & Roldán, Principios básicos de 802.11, 2006).

Pila de protocolos del 802.11.

Al igual que cualquier protocolo de la familia 802.X, el 802.11 especifica el nivel físico y el subnivel de MAC.

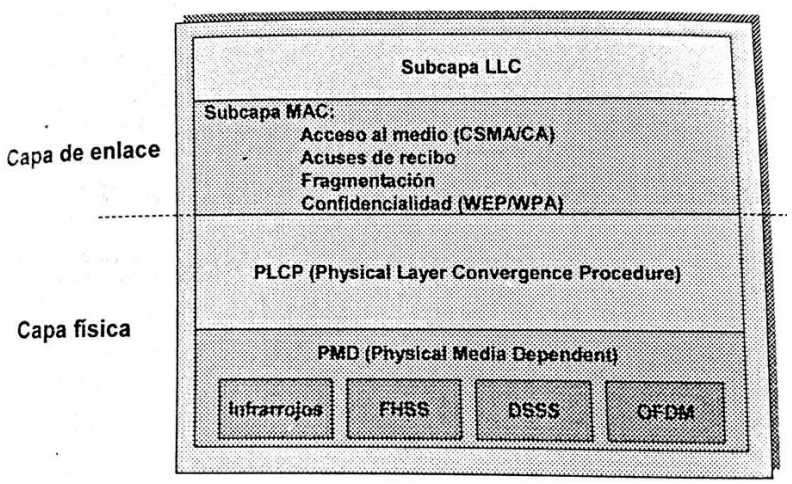


Tabla 4: Pila de protocolos 802.11. Recuperado de (Huidobro & Roldán, Pila de protocolos, 2006).

El nivel físico se encarga de resolver los aspectos relacionados con el medio de transmisión radioeléctrico. Se emplean cuatro tecnologías de transmisión diferentes, todas ellas incompatibles entre sí, las cuales son: infrarrojos, FHSS, DSSS y OFDM.

Los infrarrojos trabajan en una banda de frecuencia de 850 a 950 nm., posee velocidades de 1 y 2 Mbps, tiene un alcance a velocidad máxima de 20 m. y no atraviesa paredes. Su utilización actualmente es rara, casi nula (Huidobro & Roldán, Pila de protocolos, 2006).

El Espectro Ensanchado por Salto de Frecuencia (Frequency Hopping Spread Spectrum, FHSS) funciona, al igual que otras tecnologías usadas en el 802.11, en la frecuencia de 2.4 GHz, alcanza velocidades de 1 y 2 Mbps y una distancia de cobertura a máxima velocidad de 150 m. Es susceptible a interferencias con Bluetooth y hornos de microondas, por lo cual se usa poco (Huidobro & Roldán, Pila de protocolos, 2006).

Dependiendo de la versión del estándar 802.11 en la cual sea utilizada, el Espectro Ensanchado por Secuencia Directa (Direct Sequence Spread Spectrum, DSSS) alcanza velocidades de entre 1 y 11 Mbps, además tiene una cobertura de 30 m. y un buen rendimiento (Huidobro & Roldán, Pila de protocolos, 2006).

Por último, está el Acceso Múltiple por División de Frecuencias Ortogonales (Orthogonal Frequency Division Multiplexing, OFDMA). Puede trabajar en 2.4 y 5 GHz a velocidades 6, 9, 12, 18, 24, 36, 48 y 54 Mbps lo cual le da un máximo rendimiento, sin embargo, su alcance es de apenas 5 m (Huidobro & Roldán, Pila de protocolos, 2006).

Sobre el nivel físico se encuentra el nivel de MAC, que se encarga de funciones como la fragmentación, el acceso al medio compartido, la retransmisión de paquetes y la seguridad en la red.

III. Sistema propuesto.

Para analizar el retraso entre el envío de la señal de control por vía Ethernet y Wi-Fi y la ejecución de la orden se optó por montar un sistema en el cual el esclavo es un brazo mecánico en configuración angular y el maestro es una página web creada específicamente para este fin. Así mismo, el operador será el encargado de ejecutar las órdenes y de realizar la retroalimentación al sistema, pues este no cuenta con una automatización. Dicho sistema está compuesto por:

- Arduino Uno.
- Ethernet shield Wiznet 5100.
- NodeMCU.
- Brazo mecánico marca Steren modelo K-680.
- Cámara web.

A continuación, se realiza una descripción de cada elemento:

3.1 Arduino UNO.

Es una placa de desarrollo basada en el microcontrolador ATmega328P. Sus características técnicas más importantes son las siguientes.

- Tensión de funcionamiento de 5 volts.
- Voltaje de entrada recomendado de entre 7 y 12 volts. Los límites para el voltaje de entrada son 6 y 20 volts.
- Catorce pines digitales para I/O, de los cuales 6 pueden proporcionar una salida PWM.
- Seis pines para entrada analógica.
- La corriente DC que cada pin I/O puede proporcionar es de 20 mA.
- Memoria flash de 32 kb.
- Memoria SRAM de 2 kb.
- EEPROM de 1kb.
- Velocidad de reloj de 16 MHz.

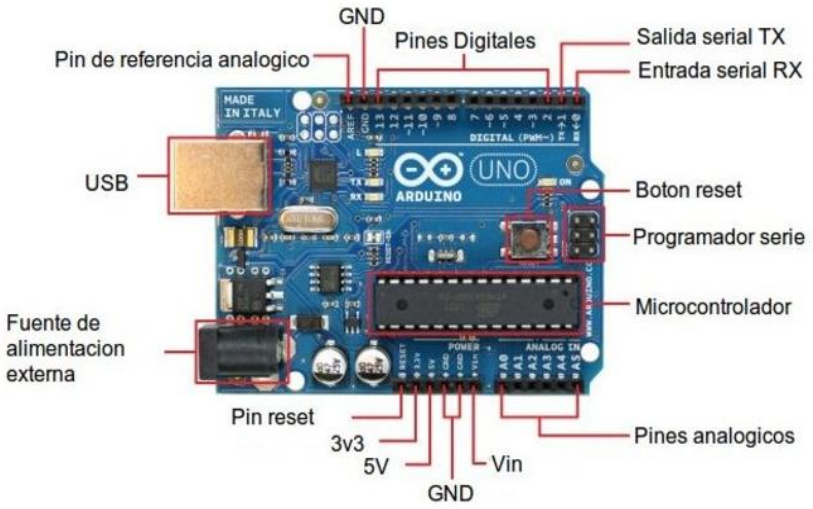


Figura 23: Diagrama de Arduino UNO. Recuperado de (INFOOTEC, s.f.).

Esta placa tiene la ventaja de programarse utilizando el IDE de Arduino, que es una plataforma de código abierto. El lenguaje utilizado es bastante similar a C/C++, aunque también es posible utilizar lenguaje de bajo nivel, por lo cual la programación y la depuración del código se vuelve más sencilla.

Además de lo anterior, es posible encontrar una amplia documentación acerca de Arduino, ejemplos de aplicaciones y proyectos realizados en dicha plataforma, blogs, comunidades, etc.

Por último, las librerías, ejemplos y la mayoría de documentación electrónica que podemos encontrar son totalmente gratis, por lo que se vuelve más fácil acceder a ella.

Dentro del sistema propuesto, Arduino tiene la función de recibir las órdenes del operador, procesarlas y ordenar al brazo mecánico la ejecución de una acción dependiendo de la información recibida. También es el encargado de almacenar tanto el código de control del brazo mecánico como la página web, es decir, actuara como servidor junto al Ethernet shield Wiznet 5100.

3.2 Ethernet shield Wiznet 5100.

Las shields son placas de circuitos modulares pueden apilarse sobre Arduino o sobre otros shields, de forma que nos permite ampliar el hardware/capacidades de Arduino (Crespo, 2015).

El Ethernet shield Wiznet 5100 permite a Arduino conectarse a Internet, es compatible con Arduino UNO y Arduino Mega. Incluye un slot para tarjeta SD, la

cual se puede utilizar para almacenar archivos que se compartirán en la red o para almacenar códigos necesarios para que los sitios web funcionen correctamente. Además, esta versión incluye un controlador de reset que asegura que el módulo inicie correctamente al energizarlo.

Otras características técnicas importante son (Arduino, s.f.):

- Voltaje de operación: 5 volts.
- Velocidad Ethernet: 10/100 Mbps.
- Interface SPI.
- Compatibilidad con TCP/UDP.
- Conector RJ45
- Librería incluida dentro del IDE de Arduino.
- Compatibilidad con IPv4.

Al usar este shield los pines 11, 12 y 13 quedan deshabilitados pues son utilizados por Arduino para comunicarse con el shield.

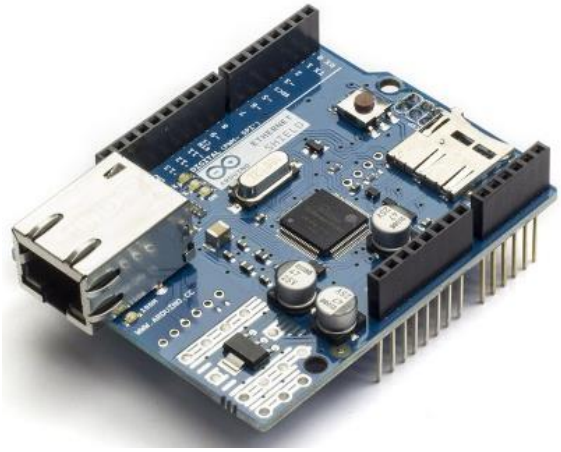


Figura 24: Ethernet Shield Wiznet 5100. Recuperado de (Arduino, s.f.).

De lo anterior es posible observar que el W5100 tendrá la función de conectar nuestro Arduino a Internet mediante Ethernet.

Este shield se considera obsoleto en comparación con otros más modernos, sin embargo, se optó por su utilización debido a que existe una buena cantidad de información en la red y ofrece las especificaciones necesarias para cumplir con los objetivos propuestos.

3.3 NodeMCU.

Es un kit de desarrollo basado en el chip ESP8266, el cual es un chip Wi-Fi de bajo costo que funciona mediante el protocolo TCP/IP, dicho chip incluye un microcontrolador Tensilica Xtensa LX106 que maneja el software necesario para la conexión 802.11.

Las características técnicas del ESP8266 son (Laborda, 2016):

- CPU de 32 bits trabajando a 80 MHz, con la posibilidad de elevarse hasta 160 MHz si se requiere.
- 64 kb de RAM para instrucciones y 96 kb de RAM para datos.
- Convertidor Analógico-Digital de 10 bits.
- 16 pines GPIO (General Purpose Input/Output, Entrada/Salida de Propósito General).

Además de lo anterior, NodeMCU tiene conexión USB al igual que Arduino, también dispone de un regulador para el voltaje de alimentación y de una memoria Flash de 1 Mb. Por si eso fuera poco, es posible programarlo usando el IDE de Arduino, el cual ya incluye las librerías necesarias para su programación.

Por lo anterior, dentro de nuestro proyecto el módulo NodeMCU realizará las funciones que realizan Arduino UNO y Ethernet Shield Wiznet 5100 pero utilizando conectividad Wi.Fi.

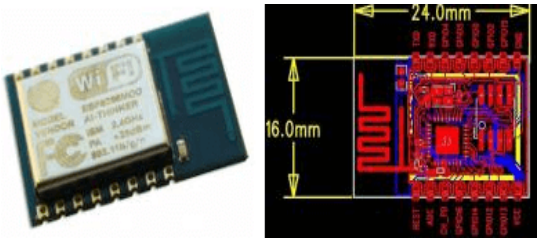


Figura 25: ESP8266. Recuperado de (Laborda, 2016).

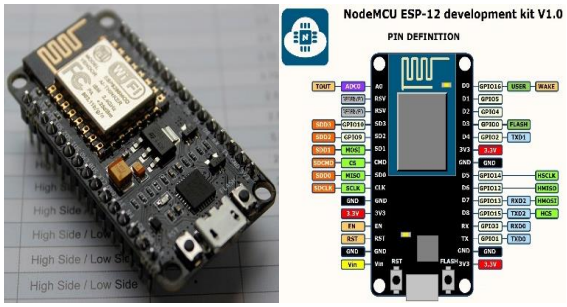


Figura 26: NodeMCU. Recuperado de (Laborda, 2016).

3.4 Brazo mecánico marca Steren, modelo K-680.

El K-680 es un modelo brazo mecánico que tiene como objetivo enseñar cuestiones básicas de robótica. El kit es bastante sencillo de ensamblar y no requiere del uso de soldadura, además de ser relativamente pequeño y ligero, lo que lo hace fácil de transportar.

Figura 27: Brazo K-680 utilizado en el proyecto. Elaboración propia.

Funciona con 6 volts DC e incluye un control remoto para manejar los 5 movimientos que puede realizar el robot y para encender el led que se encuentra ubicado en la pinza, no obstante, en el presente trabajo se ha prescindido del uso de dicho control pues los comandos serán enviados a través de una computadora.

Cuenta con una configuración angular con 4 grados de libertad, los cuales coinciden con 3 articulaciones (hombro, codo, muñeca) y una base. También está equipado con una pinza que es capaz de cargar hasta 100 g.

La libertad de movimiento que permite cada articulación, así como la capacidad de apertura de la pinza, es la siguiente (Steren):

- Base: 270°
- Hombro: 180°
- Codo: 300°
- Muñeca: 120°
- Apertura de la pinza: 4.5 cm o 1.77’’

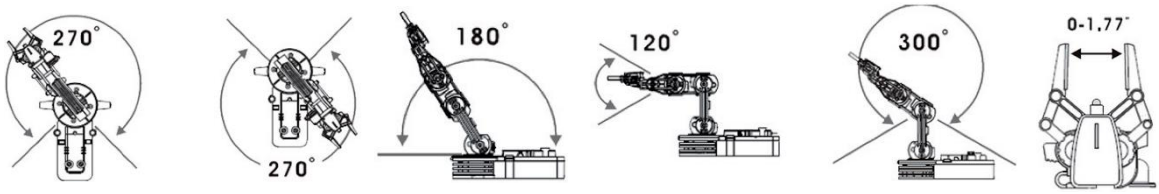


Figura 28: Libertad de movimiento del mecanismo. Recuperado de (Steren).

Para poder prescindir del control remoto incluido en el kit fue necesario fabricar un sencillo circuito electrónico que nos permitiera controlar el movimiento del brazo mediante la respuesta a señales enviadas por computadora, dicho circuito será tratado más adelante.

3.5 Cámara web.

Figura 29: Cámara web utilizada en el proyecto. Elaboración propia.

Una cámara web es un periférico de entrada que tiene como objetivo capturar audio y secuencias de movimiento, los cuales se sincronizan y codifican para ser enviados en tiempo real a través de Internet.

La cámara que nosotros utilizaremos será una Quickcam PRO 9000 de la marca Logitech. No hay un motivo específico detrás de esta decisión, sin embargo, se puede resaltar su calidad de video en HD.

Este periférico tiene una vital importancia dentro del sistema pues es el medio utilizado para monitorear la posición del mismo y su correcto funcionamiento.

3.6 Circuito electrónico para control del sentido de giro.

Una vez que se ha dado una descripción de las partes que conforman el sistema procederemos a explicar el funcionamiento del mismo.

Las conexiones y los elementos utilizados dependerán del tipo de comunicación (Ethernet o Wi-Fi) que se utilice, sin embargo, la forma en la que se controla el brazo mecánico es similar en ambos casos.

Los movimientos que realiza el K-680 son ejecutados por los motores DC instalados en cada articulación. Originalmente, dichos motores son controlados mediante

señales eléctricas enviadas por el control incluido, no obstante, al dejar de utilizarlo se hizo necesario la utilización de puentes H que nos permitan controlar el giro en un sentido u otro de cada motor. En este caso se eligió el circuito integrado L293D ya que con él es posible controlar 2 motores de manera independiente, además de funcionar con un voltaje de alimentación de 4.5 a 36 v y soportar hasta 1.2 A en cada pin de salida (aunque se recomienda que no exceda 1 A) (Texas Instruments, 2016).

La forma de conectar el L293D se explica a continuación:

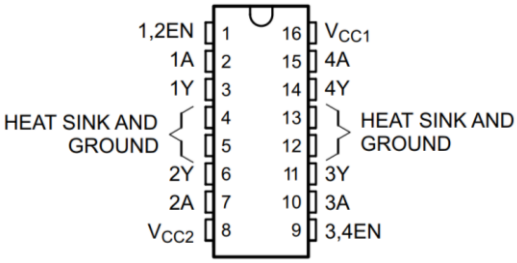


Figura 30: Configuración de pines del L293D. Recuperado de (Texas Instruments, 2016).

- Pin 1: “Enable” de los canales 1 y 2, debe ser conectado a un voltaje de un 1 lógico.
- Pines 2, 7, 10 y 15: Son las entradas a cada puente H, a estos pines se deben enviar las señales de control para controlar el sentido de giro de cada motor.
- Pines 3, 6, 11 y 14: Salidas de cada puente H, se conectan a los polos de los motores. Los pines 3 y 6 son las salidas correspondientes a las entradas 2 y 7, así mismo, los pines 11 y 14 son las salidas que corresponden a las entradas 10 y 15.
- Pines 4, 5, 12 y 13: Deben ser conectados GND.
- Pin 8: Entrada de alimentación VCC. Soporta entre 4.5 y 36 v.
- Pin 9: “Enable” de los canales 3 y 4, al igual que el pin 1 debe ser conectado a un voltaje de un 1 lógico.
- Pin 16: Entrada de voltaje lógico de 5 v.

Para controlar los 5 motores del brazo mecánico se fabricó con ayuda del software EAGLE un circuito electrónico formado por 3 encapsulados L393D, el cual se muestra en la siguiente imagen.

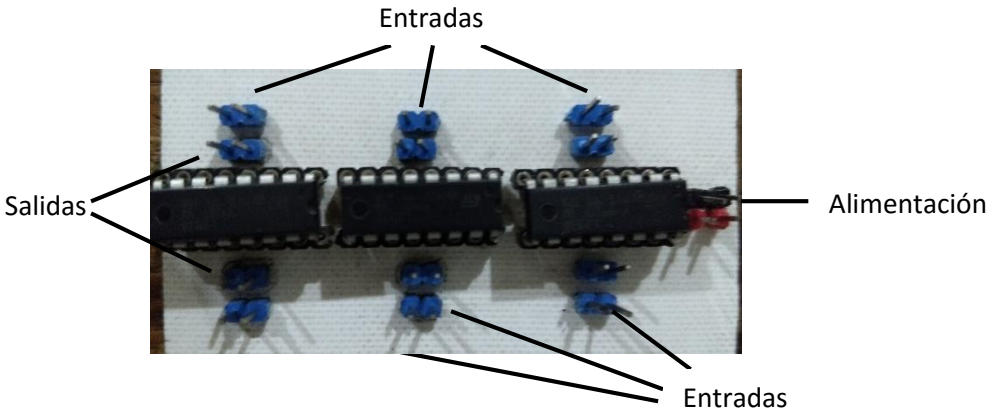


Figura 31: Circuito electrónico. Elaboración propia

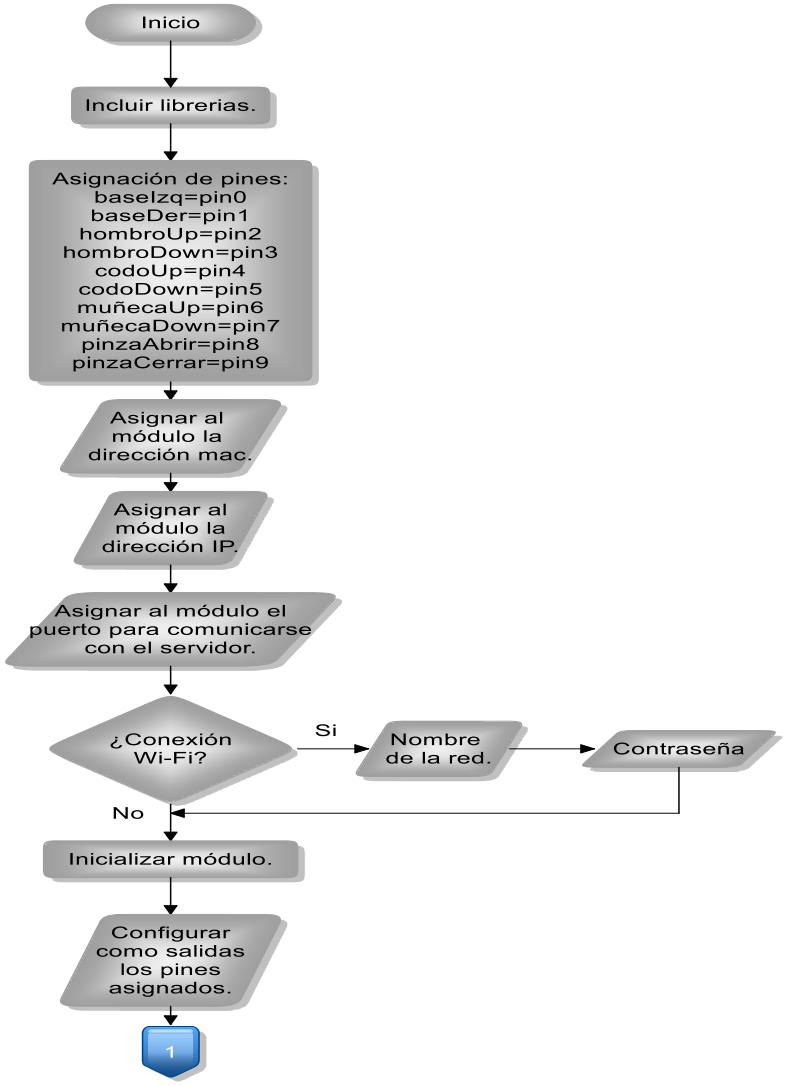
Ya que el sistema puede trabajar con 5 volts los pines 1, 8, 9 y 16 se conectaron en serie para ser alimentados con 5 volts.

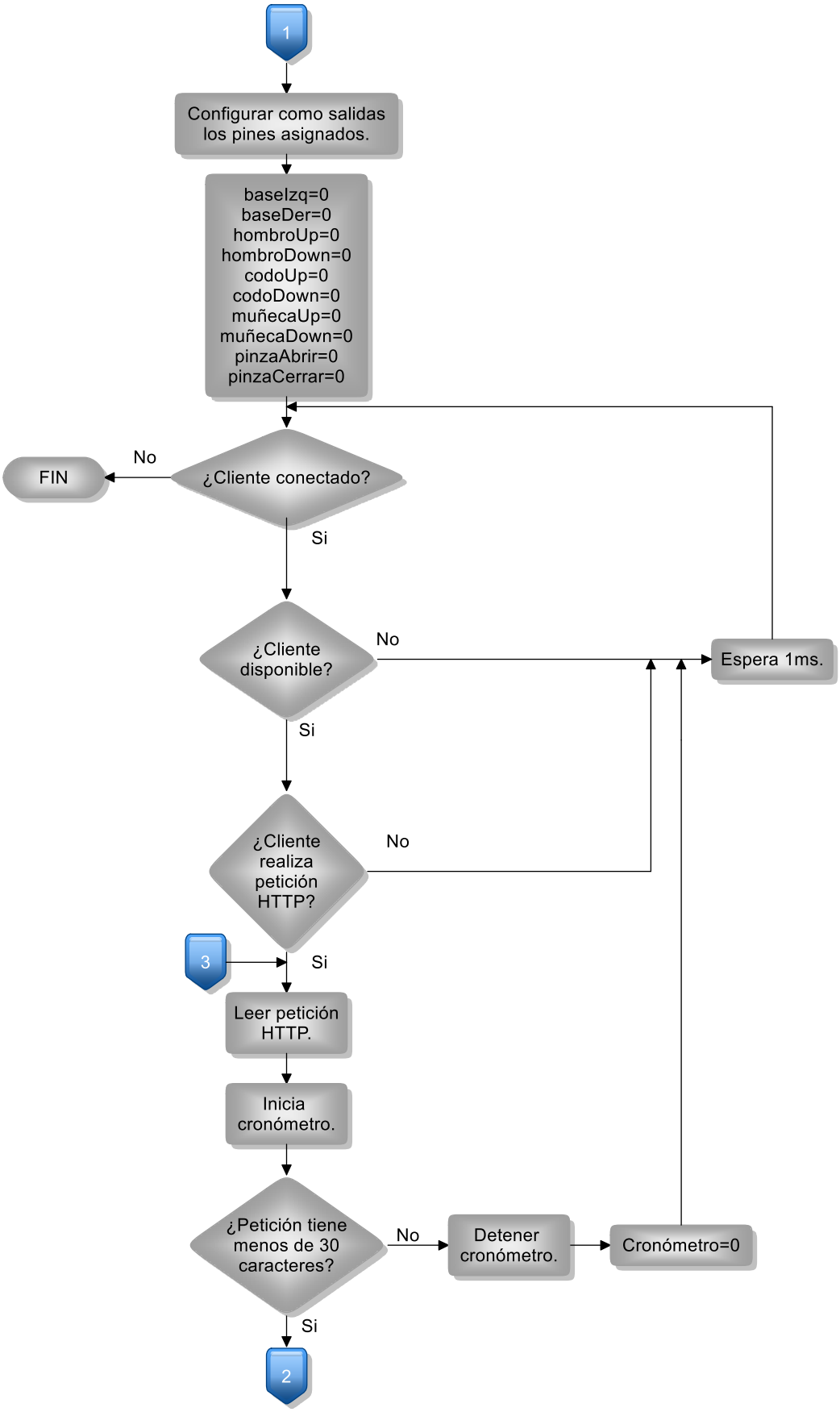
Tanto para la conexión Ethernet como para la conexión Wi-Fi los pines de salida utilizados van del 0 al 9.

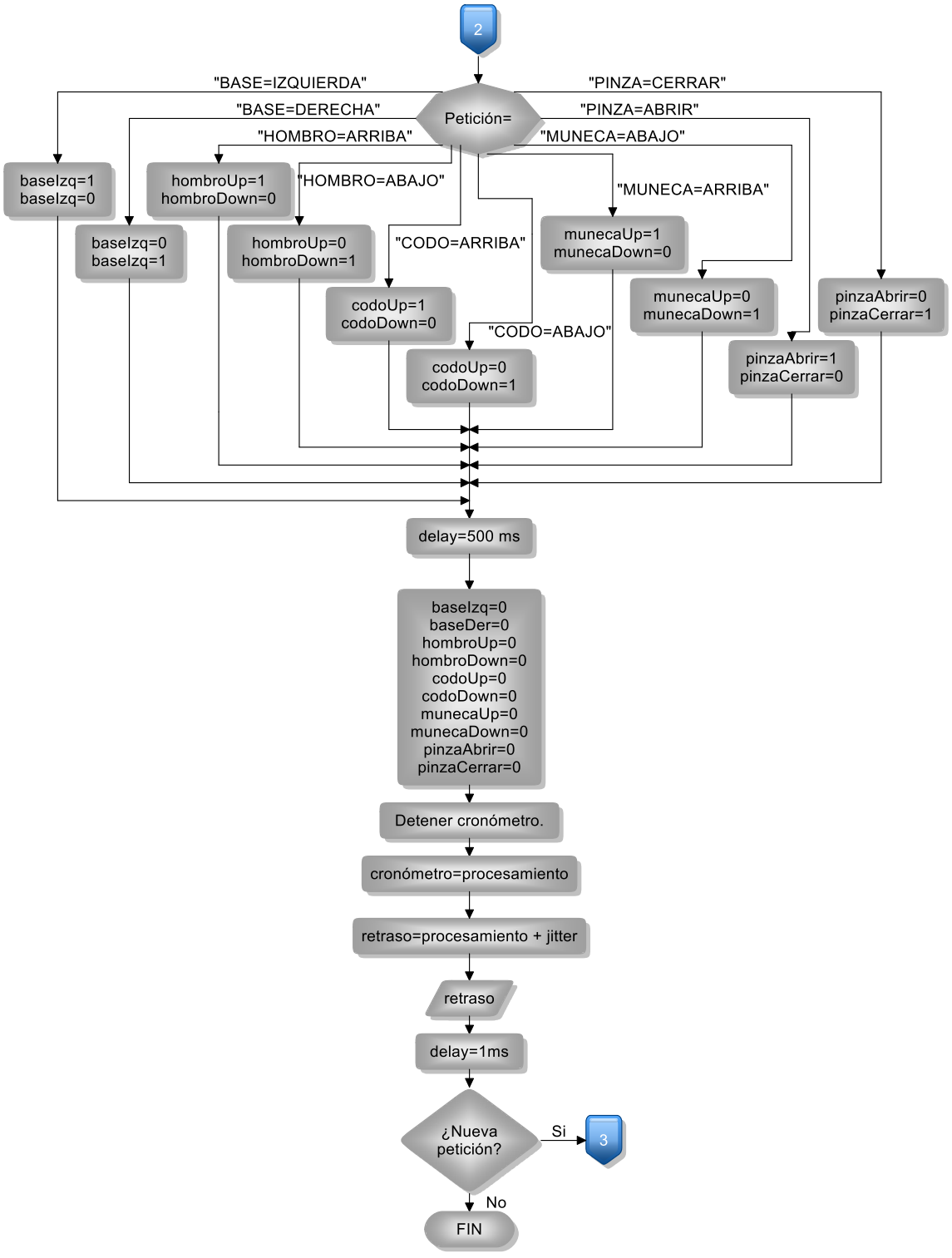
3.7 Diagrama de flujo.

Definiremos diagrama de flujo como la manera de representar gráficamente un proceso o algoritmo a través de una serie de pasos estructurados y vinculados.

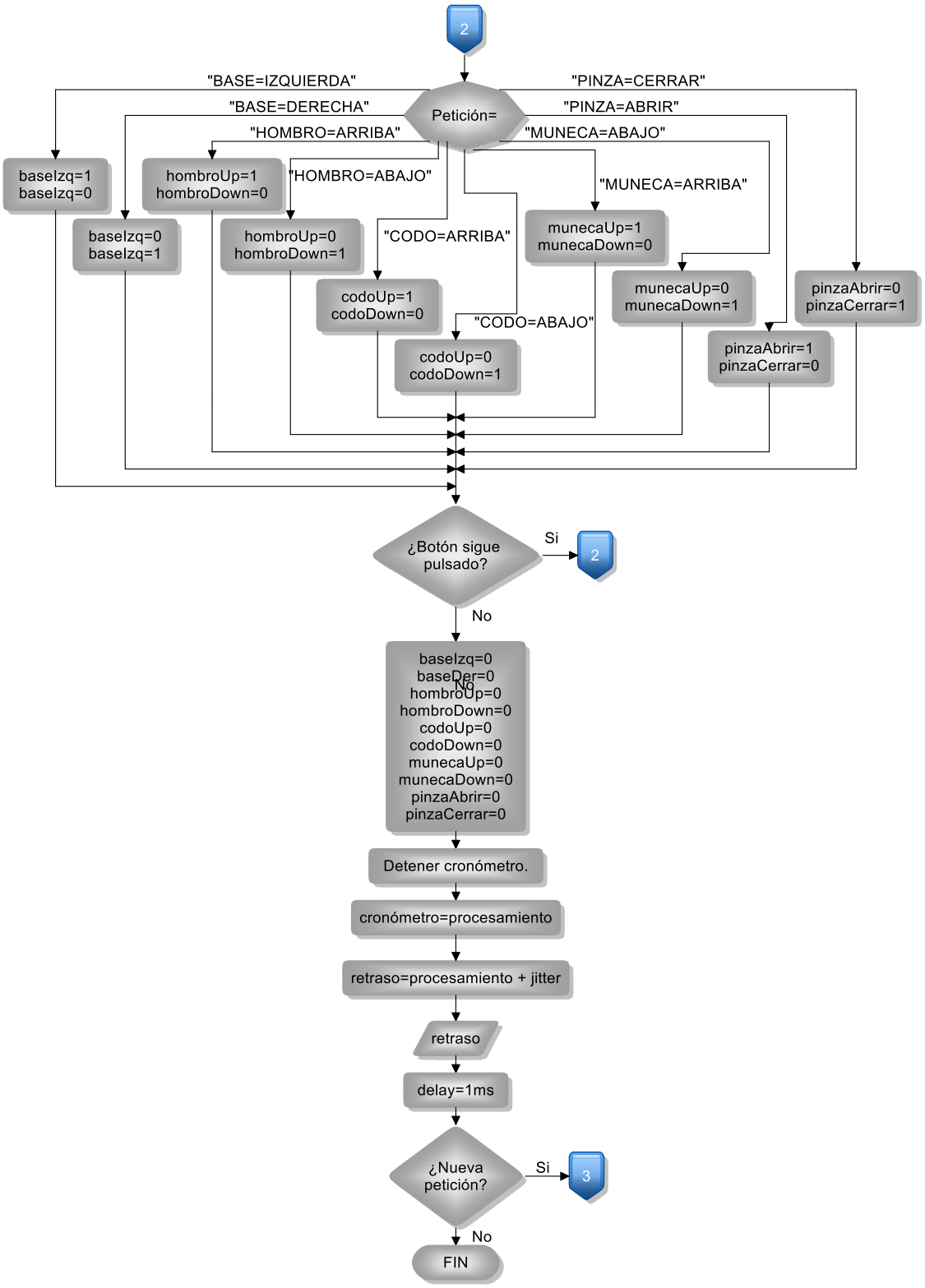
A pesar de que se utilizaron diferentes placas de desarrollo y de que su funcionamiento y configuración es diferente se puede decir que el diagrama de flujo es igual para ambos. Las diferencias de funcionamiento y de configuración serán tratadas más adelante.




1
Configurar como salidas los pines asignados.
baseIzq=0
baseDer=0
hombroUp=0
hombroDown=0
codoUp=0
codoDown=0
muñecaUp=0
muñecaDown=0
pinzaAbrir=0
pinzaCerrar=0
¿Cliente conectado?
No
FIN
Si
¿Cliente disponible?
No
Espera 1ms.
Si
¿Cliente realiza petición HTTP?
No
3
Si
Leer petición HTTP.
Inicia cronómetro.
¿Petición tiene menos de 30 caracteres?
No
Detener cronómetro.
Cronómetro=0
Si
2


2
"BASE=IZQUIERDA"
"BASE=DERECHA"
"HOMBRO=ARRIBA"
"HOMBRO=ABAJO"
"CODO=ARRIBA"
"CODO=ABAJO"
"MUNECA=ARRIBA"
"MUNECA=ABAJO"
"PINZA=CERRAR"
"PINZA=ABRIR"
Petición=
baseIzq=1
baseIzq=0
baseIzq=0
baseIzq=1
hombroUp=1
hombroDown=0
hombroUp=0
hombroDown=1
codoUp=1
codoDown=0
codoUp=0
codoDown=1
munecaUp=1
munecaDown=0
munecaUp=0
munecaDown=1
pinzaAbrir=0
pinzaCerrar=1
pinzaAbrir=1
pinzaCerrar=0
delay=500 ms
baseIzq=0
baseDer=0
hombroUp=0
hombroDown=0
codoUp=0
codoDown=0
munecaUp=0
munecaDown=0
pinzaAbrir=0
pinzaCerrar=0
Detener cronómetro.
cronómetro=procesamiento
retraso=procesamiento + jitter
retraso
delay=1ms
¿Nueva petición?
Si
3
No
FIN

Para realizar un análisis más profundo sobre los tiempos de retraso se crearon 2 versiones del programa. Ambas versiones se rigen prácticamente por el mismo diagrama de flujo, la principal diferencia es la siguiente:



3.8 El código.

A continuación, se explicarán las partes más importantes de las diferentes versiones del código que fueran realizadas. En la sección de apéndices se pueden encontrar los códigos en su totalidad.

Debido a que el W5100 y el NodeMCU realizan la función de servidor y de cliente, la página web se encuentra dentro del código, por lo tanto, también se hablara de ella en este capítulo.

3.8.1 Ethernet v1.

El shield W5100 nos permite utilizar los pines del 0 al 9 como entradas o como salidas, por lo tanto, podemos utilizarlas para asignar las salidas sin interferir con el funcionamiento general del módulo.

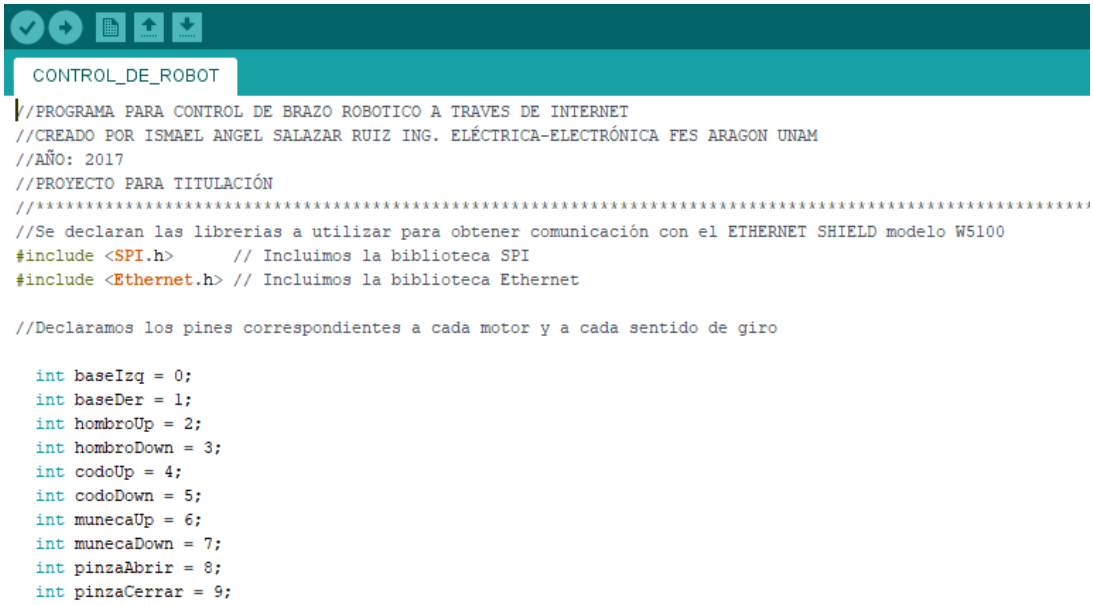


Figura 32: Inicio del código para control de brazo mecánico vía Ethernet. Elaboración propia.

En la Figura 32 se puede apreciar la inclusión de la librería <Ethernet.h> y <SPI.h>. La primera contiene las funciones necesarias para conectarse a Internet por Ethernet tanto en modo cliente como en modo servidor, además de permitir la utilización de pseudocódigo para páginas web (HTML, CSS, JavaScript, etc.), a su vez, la librería <SPI.h> se utiliza para comunicarse con la tarjeta microSD, aunque esta no es estrictamente necesaria para el proyecto y se puede prescindir de su uso.

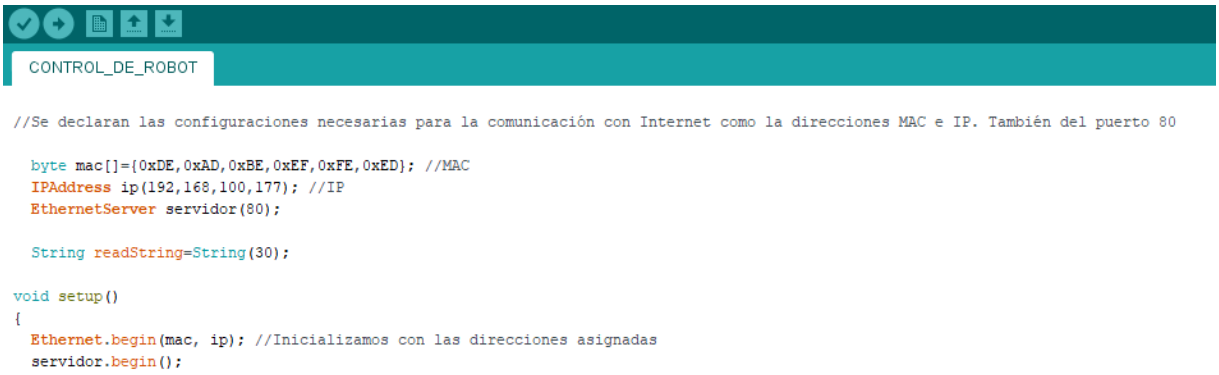


Figura 33: Algunas configuraciones necesarias para la comunicación Ethernet. Elaboración propia.

Cada módulo tiene una dirección MAC predefinida que es posible modificar mediante el comando byte mac [], siempre prestando atención a que la nueva mac no coincida con alguna que ya este asignada en la red.

El comando IPAddress nos permite asignar de manera manual una dirección IP al módulo, sin embargo, no es estrictamente necesario que asignemos la dirección IP de esta manera pues nuestro router le asignara mediante DHCP una dirección IP a cada dispositivo que se conecte a la red.

La siguiente línea, EthernetServer servidor (80), asigna el puerto por el cual nos comunicaremos con el servidor, por defecto el puerto elegido será el 80, al igual que con los parámetros anteriores, podemos cambiar el puerto por alguno que no entre en conflicto con las configuraciones de nuestra red.

La variable String readString=String (30) será la encargada de almacenar las peticiones realizadas por el cliente, dicha petición será una cadena de caracteres con una longitud no mayor a 30 caracteres.

Dentro del setup iniciamos el modulo Ethernet con los parámetros mac e ip. Posteriormente iniciamos el servidor web mediante servidor. begin ().

En la figura 38 observamos que la instrucción EthernetClient cliente= servidor. available () nos permite conectarnos como cliente siempre y cuando el servidor esté disponible. Posteriormente se agregan una serie de condiciones que van encaminadas a comprobar si el cliente se encuentra conectado y disponible (cliente. connected () y cliente. available (), respectivamente). Una vez que se comprueba que el cliente cumple con dichos requisitos se procede a leer la petición realizada por el mismo y a comprobar que dicha petición tiene menos de 30 caracteres mediante la instrucción readString.length() <30. Por último, se crean variables de

tipo entero que serán utilizadas para que el programa identifique cual motor debe moverse.

Figura 34: Loop del código. Elaboración propia.

Es necesario aclarar que dichas peticiones aparecerán en la barra de búsqueda y desde ahí serán leídas e interpretadas por el servidor.

```
//Movimiento de la base

        if(readString.substring(BASE,BASE+14)=="BASE=IZQUIERDA")
        {
           digitalWrite (baseIzq,HIGH);
           digitalWrite (baseDer,LOW);
           delay(500);
           digitalWrite (baseIzq,LOW);
           digitalWrite (baseDer,LOW);
           delay(1);
        }
        else if (readString.substring(BASE,BASE+12)=="BASE=DERECHA")
        {
           digitalWrite (baseIzq,LOW);
           digitalWrite (baseDer,HIGH);
           delay(500);
           digitalWrite (baseIzq,LOW);
           digitalWrite (baseDer,LOW);
           delay(1);
        }
```

Figura 35: Ejemplo de condiciones necesarias para realizar los movimientos. Elaboración propia.

En la figura 35 se ejemplifica, mediante los movimientos de la base, las subrutinas utilizadas para controlar los movimientos del brazo mecánico. Es fácil observar que

para decidir cuál movimiento realizar se utilizó la estructura if por considerarse más fácil de interpretar.

La instrucción readString.substring(BASE, BASE+14) =="BASE=IZQUIERDA" funciona identificando la cadena de caracteres que se encuentra del lado izquierdo de la coma, una vez identificada dicha cadena se procede a realizar un cálculo mediante suma para comprobar si coincide con alguno de los movimientos programados, dicho de otra manera, lo primero que hace es identificar la palabra BASE, posteriormente se comienza a leer de izquierda a derecha el número de caracteres indicados por la suma, en este caso 14 (si se deseara realizar la lectura de derecha a izquierda la estructura debe cambiar y en lugar de realizarse una suma se realiza una resta), posteriormente se comprueba si la petición coincide con la instrucción "BASE=IZQUIERDA". Aunque las cadenas de caracteres cambian dependiendo del movimiento a realiza, el funcionamiento de la instrucción es el mismo.

Una vez que se ha identificado el movimiento a realizar, uno de los pines del motor correspondiente será puesto en 1 y el otro pin será puesto en 0 durante 500 ms, después de ese tiempo ambos pines se pondrán en 0 para detener el movimiento. Se espera 1 ms para permitir que el servidor esté listo para recibir una nueva instrucción.

3.8.2 Página web Ethernet v1.

Para poder incluir pseudocódigo HTML en el w5100 se debe de incluir la instrucción client. printl. A continuación, se anexa una imagen que muestra una parte del código necesario para la página web.

```
cliente.println("HTTP/1.1 200 OK");
cliente.println("Content-Type: text/html;charset=utf-8");
cliente.println();
//Página Web en HTML
cliente.println("<html>");
cliente.println("<head>");
cliente.println("<title>PANEL DE CONTROL BRAZO ROBÓTICO</title>");
cliente.println("</head>");
cliente.println("<body width=100% height=100%>");
cliente.println("<center>");
cliente.println("<h1>FES ARAGÓN</h1>");
cliente.println("<h1>PANEL DE CONTROL BRAZO ROBÓTICO</h1>");
cliente.print("<br><br>");
cliente.print("Base: ");
cliente.print("<br>");
cliente.println(F("<input type=submit value=← style=width:200px;height:75px onClick=location.href='./?BASE=IZQUIERDA\'>"));
cliente.println(F("<input type=submit value=→ style=width:200px;height:75px onClick=location.href='./?BASE=DERECHA\'>"));
cliente.print("<br>");
cliente.print("Hombro: ");
cliente.print("<br>");
cliente.println(F("<input type=submit value=↑ style=width:200px;height:75px onClick=location.href='./?HOMBRO=ARRIBA\'>"));
cliente.println(F("<input type=submit value=↓ style=width:200px;height:75px onClick=location.href='./?HOMBRO=ABAJO\'>"));
```

Figura 36: Muestra de la programación de la página web. Elaboración propia.

En la figura 36 se puede observar que las etiquetas (html, head, title, etc.) se colocan entre los símbolos de *menor qué (<)* y *mayor que,* esto se hace para indicarle a

Arduino que comenzaremos a utilizar pseudocódigo html. Ya se mencionó que es posible utilizar JavaScript, CSS o AJAX para realizar una programación más llamativa y amigable, sin embargo, se corre el riesgo de sobrecargar el servidor y alentar la comunicación.

Otro punto importante a tratar es la configuración de cada botón. Tomaremos como ejemplo la siguiente línea de código:

```
<input type=submit value=← style=width:200px;height:75px
onClick=location.href='./?BASE=IZQUIERDA\'>
```

Al declarar “input type=submit” se inicia un elemento tipo submit, es decir, un botón, después se introduce el texto que aparecerá dentro del mismo mediante “value=←”.

La instrucción “style=width:200px;height:75px” sirve para declarar los atributos físicos que tendrá el botón, aquí pueden modificarse cosas como el color, la forma y el tamaño, en este caso se le asigna un tamaño de 200 pixeles de ancho por 75 de alto.

Por último tenemos la parte más importante de este elemento, la instrucción “onClick=location.href='./?BASE=IZQUIERDA\'”, onClick se refiere al tipo de evento que desarrollara en el botón, en este caso, al hacer clic izquierdo sobre él se disparara el evento “location.href=” que se encarga de imprimir en la barra de búsqueda el texto “./?BASE=IZQUIERDA\”. El aspecto de la página web es el siguiente (el aspecto de la página no cambia para Ethernet o Wi-Fi):

Figura 37: Página web utilizada para la primera versión del código Ethernet. Elaboración propia.

3.8.3 Wi-Fi v1.

Para poder utilizar el NodeMCU es necesario conocer su diagrama de pines, el cual se muestra en la figura 38:

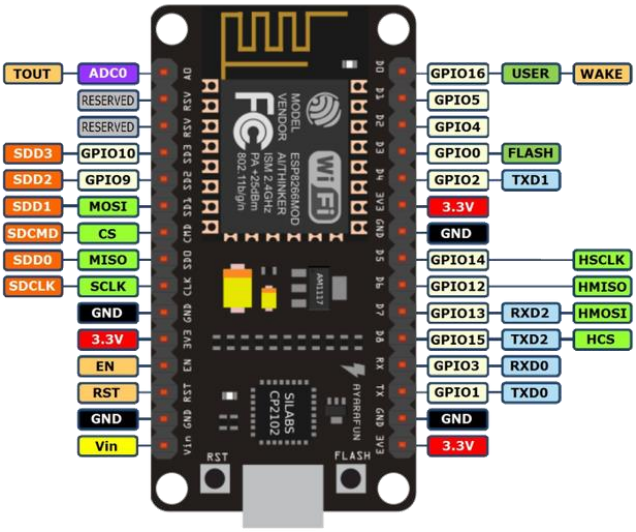


Figura 38: Diagrama de pines del NodeMCU. Recuperado de (Singh, 2016).

En este caso al configurar los pines de salida dentro del código elegiremos las GPIO 16, 5, 4, 0, 2, 14, 12, 13, 15 y 3 ya que corresponden a los pines 0, 1, 2, 3, 4, 5, 6, 7, 8 y 9 respectivamente.

Podemos apreciar en la figura 39 que fue necesario seleccionar el tipo de placa antes de compilar y subir el archivo.

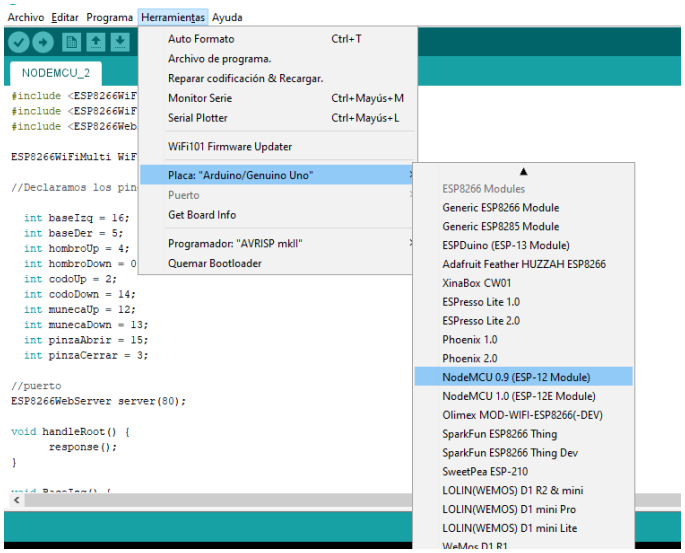


Figura 39: Cambio de placa en el IDE de Arduino. Elaboración propia.

Después de seleccionar el NodeMCU en el gestor de placas podemos explicar el código.

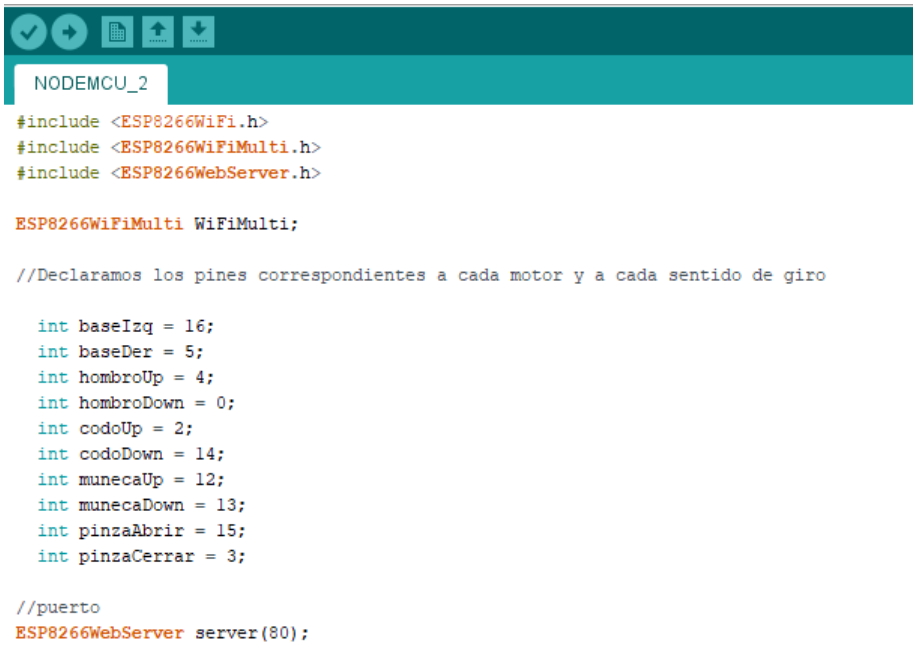


Figura 40: Configuración inicial del código. Elaboración propia.

La primera librería nos permite cargar las instrucciones y comandos necesarios para conectar el ESP8266 a Internet. La segunda permite conectar el módulo a varias redes de manera que, si se pierde la conexión en una pasa a la siguiente, se eligió esta librería por ofrecer mayor comodidad a la hora de configurarse. Por último, la tercera librería nos permite utilizar el ESP8266 como servidor web. En las siguientes líneas podemos observar la declaración de pines que fue mencionada anteriormente. Finalmente seleccionamos el puerto de comunicación.

Figura 41: Algunas subrutinas para control de movimiento. Elaboración propia.

En esta ocasión se optó por utilizar subrutinas para cada movimiento del robot, esto debido a los comandos utilizados por el NodeMCU para realizar peticiones desde el modo cliente. Podemos observar que el funcionamiento de dichas subrutinas es igual que en w5100 excepto que en este caso se incluyó la subrutina “void handleRoot ()” que nos permite regresar al documento raíz de la página web para esperar otra petición una vez realizada la subrutina de movimiento correspondiente.

Una de las ventajas que se presentan al utilizar este módulo es la reducción en la complejidad necesaria para agregar el pseudocódigo de la página web. Como se aprecia en la figura 42.

```
const String HtmlHtml = "<html><head><title>PANEL DE CONTROL BRAZO ROBÓTICO</title></head>";
const String HtmlHtmlClose = "</html>";
const String HtmlTitle = "<h1><CENTER>FES ARAGÓN</CENTER></h1>";
const String Htmlbody = "<body width=100% height=100%>";
const String HtmlbodyClose = "</body>";
const String HtmlTituLo = "<h1><CENTER>PANEL DE CONTROL DE BRAZO ROBÓTICO</h1><br>";

const String HtmlBaseIzquierda = "<CENTER>Base:<br><input type=submit value=← style=width:200px;height:75px onClick=location.href='BASE=IZQUIERDA\'>";
const String HtmlBaseDerecha = "<input type=submit value=→ style=width:200px;height:75px onClick=location.href='BASE=DERECHA\'><br>";

const String HtmlHombroArriba = "<CENTER>Hombro:<br><input type=submit value=↑ style=width:200px;height:75px onClick=location.href='HOMBRO=ARRIBA\'>";
const String HtmlHombroAbajo = "<input type=submit value=↓ style=width:200px;height:75px onClick=location.href='HOMBRO=ABAJO\'><br>";

const String HtmlCodoArriba = "<CENTER>Codo:<br><input type=submit value=↑ style=width:200px;height:75px onClick=location.href='CODO=ARRIBA\'>";
const String HtmlCodoAbajo = "<input type=submit value=↓ style=width:200px;height:75px onClick=location.href='CODO=ABAJO\'><br>";

const String HtmlMunecaArriba = "<CENTER>Muñeca:<br><input type=submit value=↑ style=width:200px;height:75px onClick=location.href='MUNECA=ARRIBA\'>";
const String HtmlMunecaAbajo = "<input type=submit value=↓ style=width:200px;height:75px onClick=location.href='MUNECA=ABAJO\'><br>";

const String HtmlPinzaAbrir = "<CENTER>Pinza:<br><input type=submit value=ABRIR style=width:200px;height:75px onClick=location.href='PINZA=ABRIR\'>";
const String HtmlPinzaCerrar = "<input type=submit value=CERRAR style=width:200px;height:75px onClick=location.href='PINZA=CERRAR\'>";
```

Figura 42: Pseudocódigo en NodeMCU. Elaboración propia.

Ahora para agregar alguna línea de pseudocódigo basta con declararla como una variable de tipo “const String”. Para ensamblar toda la página se crea una subrutina de la siguiente manera:

```
void response(){
  String htmlRes=HtmlHtml+HtmlTitle+HtmlTituLo+Htmlbody+HtmlbodyClose+HtmlBaseIzquierda+HtmlBaseDerecha+HtmlHombroArriba+HtmlHombroAbajo+HtmlCodoArriba+HtmlCodoAbajo+HtmlMunecaArriba+HtmlMunec

 server.send(200, "text/html;charset=utf-8", htmlRes);
}
```

Figura 43: Subrutina para la página web. Elaboración propia.

En donde “String htmlRes” es la variable que se encarga de unir toda la página realizando una suma de todas las variables “const String” creadas anteriormente. Esa información es enviada al servidor junto con el tipo de texto utilizado en la programación de la misma.

Dentro del setup encontramos configuraciones como el nombre y contraseña de la red a conectar, la inicialización del módulo y la configuración de los pines, más importante aún es la manera en la que el servidor recibe e interpreta las peticiones del cliente:

```
server.on("/", handleRoot);
server.on("/BASE=IZQUIERDA", BaseIzq);
server.on("/BASE=DERECHA", BaseDer);
server.on("/HOMBRO=ARRIBA", HombroArriba);
server.on("/HOMBRO=ABAJO", HombroAbajo);
server.on("/CODO=ARRIBA", CodoArriba);
server.on("/CODO=ABAJO", CodoAbajo);
server.on("/MUNECA=ARRIBA", MunecaArriba);
server.on("/MUNECA=ABAJO", MunecaAbajo);
server.on("/PINZA=ABRIR", PinzaAbrir);
server.on("/PINZA=CERRAR", PinzaCerrar);
```

Figura 44: Instrucciones para que el servidor reciba e interprete las peticiones del cliente. Elaboración propia.

La instrucción “server.on” mantiene activo al servidor y le permite recibir las peticiones del cliente, los parámetros que se encuentran entre paréntesis son la cadena de caracteres que espera recibir y la subrutina a la cual debe dirigirse cuando reciba dicha cadena.

```
void loop() {

  server.handleClient();

}
```

Figura 45: Servidor espera la petición del cliente. Elaboración propia.

Por último, dentro del Loop el servidor estará esperando la petición del cliente mediante server.handleClient.

Como pudimos observar la programación del NodeMCU es más amable y sencilla que la del w5100.

3. 8. 4. Ethernet y Wi-Fi v2.

Anteriormente se mencionó que para poder estudiar el tiempo de retardo en las comunicaciones se crearon 2 versiones para cada programa, la principal diferencia entre ambas radica en el tiempo que se realiza cada movimiento. En la primera versión cada movimiento se realiza durante 500 ms, en la segunda versión el tiempo depende del usuario pues los botones fueron programados para disparar un evento mientras el usuario se mantenga realizando clic izquierdo sobre él. Veamos cómo se realizó la programación tomando como ejemplo el botón que mueve la base a la izquierda:

```
cliente.println(F("<input type='button' value=\"←\" style=width:200px;height:75px
onpointerdown=\"location.href='/?BASE=IZQUIERDA';\"
onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
```

La primera modificación fue declarar directamente “type=’button’”, button es un elemento que acepta una variedad más amplia de eventos, para este caso nos interesan cuatro:

- onmousedown
- onmouseup
- onpointerdown
- onpointerup

Los cuatro tienen un funcionamiento similar, sin embargo, los eventos “onpointer” funcionan tanto en dispositivos táctiles como en los que no lo son.

En el código la instrucción “onpointerdown=\"location.href='/?BASE=IZQUIERDA';\"” nos indica que mientras el botón este pulsado el cliente enviara la cadena correspondiente al movimiento de

la base a la izquierda. Para detener el movimiento es necesario detectar cuando el botón deja de ser pulsado, eso se logra con la instrucción “onpointerup=\"location.href='/?ESCLAVO=APAGADO';\"”, cuando el programa detecta que se dejó de pulsar el botón el cliente envía una cadena que indica que se debe detener el movimiento. Ya que ahora es necesario identificar la cadena que pide que se detenga el movimiento se creó una condicionante extra:

```
else if (readString.substring(ESCLAVO,ESCLAVO+15)=="ESCLAVO=APAGADO")
 {
    digitalWrite (baseIzq,LOW);
    digitalWrite (baseDer,LOW);
    digitalWrite (hombroUp,LOW);
    digitalWrite (hombroDown,LOW);
    digitalWrite (codoUp,LOW);
    digitalWrite (codoDown,LOW);
    digitalWrite (munecaUp,LOW);
    digitalWrite (munecaDown,LOW);
    digitalWrite (pinzaAbrir,LOW);
    digitalWrite (pinzaCerrar,LOW);
 }
```

Figura 46: Condición para detener el movimiento del brazo. Elaboración propia.

Se puede observar que su funcionamiento es igual a las demás condicionales solo que en esta se ponen en 0 todos los pines.

Adicionalmente a lo anterior se observó que en ocasiones el brazo no detenía su movimiento al soltar el botón ya que la acción entre pulsa y soltar ocurría muy deprisa y la cadena de alto no era enviada, para detener el movimiento en estos casos se creó un botón que funcionara como un alto de emergencia:

```
cliente.println(F("<input type='button' value=\"Paro de emergencia\" style=width:200px;height:75px onClick=\"location.href='/?ESCLAVO=APAGADO';\">"));
```

Se programó igual que los botones de la versión anterior del código.

Con el NodeMCU la programación se realizó de la siguiente manera. Para agregar los eventos onpointer se utilizó la estructura:

```
const String HtmlBaseIzquierda = "<CENTER>Base:<br><input type='button' value=←
style=width:200px;height:75px onpointerdown=\"location.href='BASE=IZQUIERDA';\"
onpointerup=\"location.href='ESCLAVO=APAGADO';\">";
```

La subrutina agregada para detectar la petición de alto fue:

```
void Off(){
  digitalWrite (baseIzq,LOW);
  digitalWrite (baseDer,LOW);
  digitalWrite (hombroUp,LOW);
  digitalWrite (hombroDown,LOW);
  digitalWrite (codoUp,LOW);
  digitalWrite (codoDown,LOW);
  digitalWrite (munecaUp,LOW);
  digitalWrite (munecaDown,LOW);
  digitalWrite (pinzaAbrir,LOW);
  digitalWrite (pinzaCerrar,LOW);
  response();
}
```

Figura 47: Subrutina de paro. Elaboración propia.

De igual manera se agregó un botón de alto de emergencia en caso de que el brazo no se detuviera, la programación fue la siguiente:

const String HtmlOff = "<CENTER>Apagado:
<input type=submit value=Paro de emergencia style=width:200px;height:75px onClick=location.href='ESCLAVO=APAGADO\'>";

Al igual que en el caso anterior, el botón funciona con el evento onClick.

IV. Resultados.

Para medir el retraso se deben tomar en cuenta la velocidad de ejecución de las instrucciones en Arduino y en NodeMCU, la velocidad de conexión a Internet y la velocidad de respuesta del servidor.

Para la velocidad de ejecución de las instrucciones contamos con la función millis, que nos indicará el tiempo transcurrido desde el inicio del programa hasta un determinado punto del mismo. La velocidad de conexión y la velocidad de respuesta del servidor están estrechamente relacionados por lo que se puede considerar como una sola variable, para realizar una medición fiable se utilizara la función proporcionada por los buscadores Google Chrome, Mozilla Firefox y Microsoft Edge al seleccionar en la pestaña “inspeccionar elemento”. Los datos registrados por los tres buscadores serán comparados para tener mayor certeza del resultado obtenido.

En las imágenes 48, 49 y 50 se puede observar la pantalla de medición de tiempo en Google Chrome, Mozilla Firefox y Microsoft Edge respectivamente.

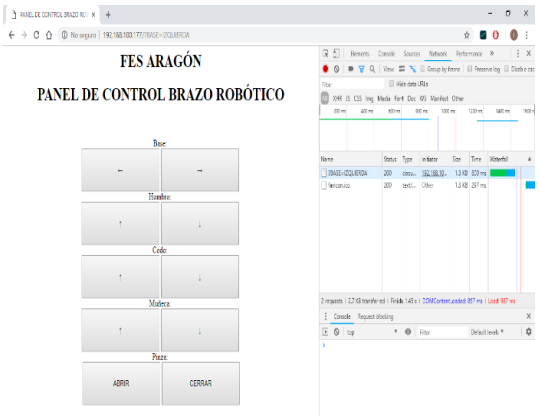


Figura 48: Inspeccionar elemento en Google Chrome. Elaboración propia.

Figura 49: Inspeccionar elemento en Mozilla Firefox. Elaboración propia.

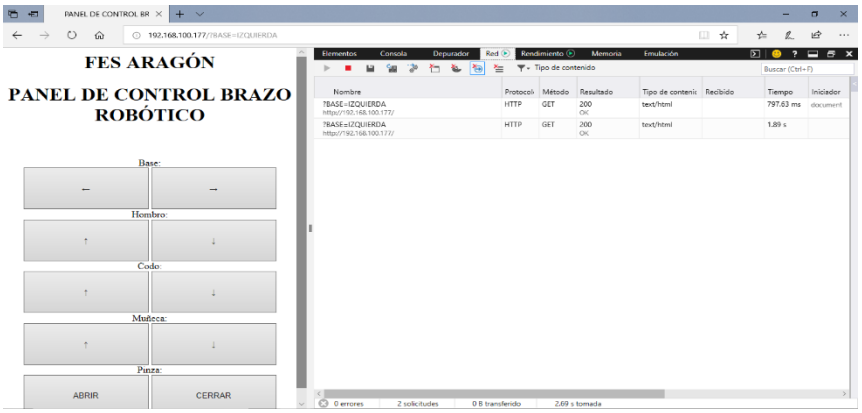


Figura 50: Inspeccionar elemento en Microsoft Edge. Elaboración propia.

Utilizando la función millis nos fue posible saber que el tiempo de procesamiento de Arduino es de 500 ms, mientras que para el NodeMCU fue de 400 ms, por lo tanto, el tiempo de retardo desde que el operador da clic sobre un botón hasta el momento en el que es realizada la instrucción será el valor proporcionado por el buscador más 500 en el caso de Arduino y más 400 para NodeMCU.

A continuación, se mostrarán los resultados obtenidos de dichas mediciones. Los tiempos de retraso (Delay) se muestran en milisegundos.

4. 1 Red LAN

4. 1. 1 Ethernet v1.

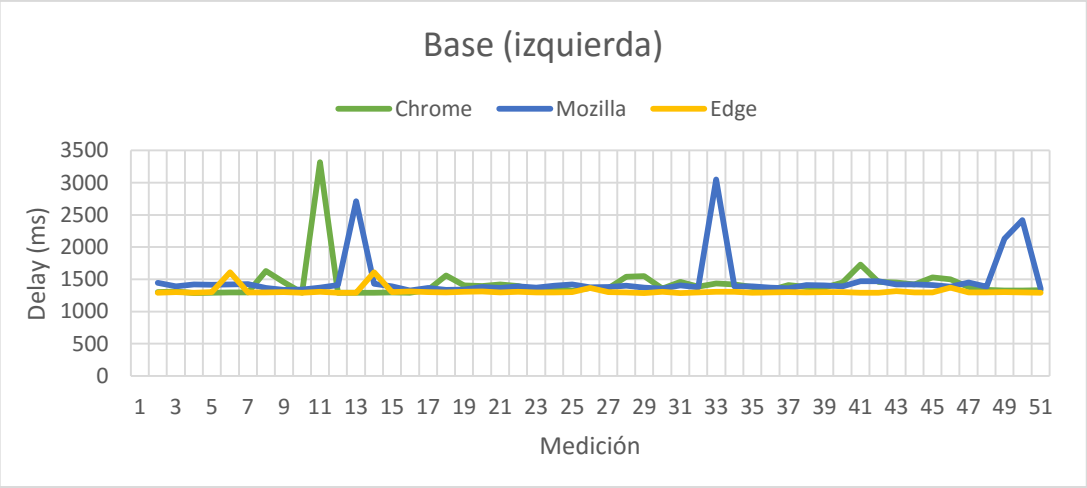


Figura 51: Movimiento de la base hacia la izquierda, red LAN, Ethernet v1. Elaboración propia.

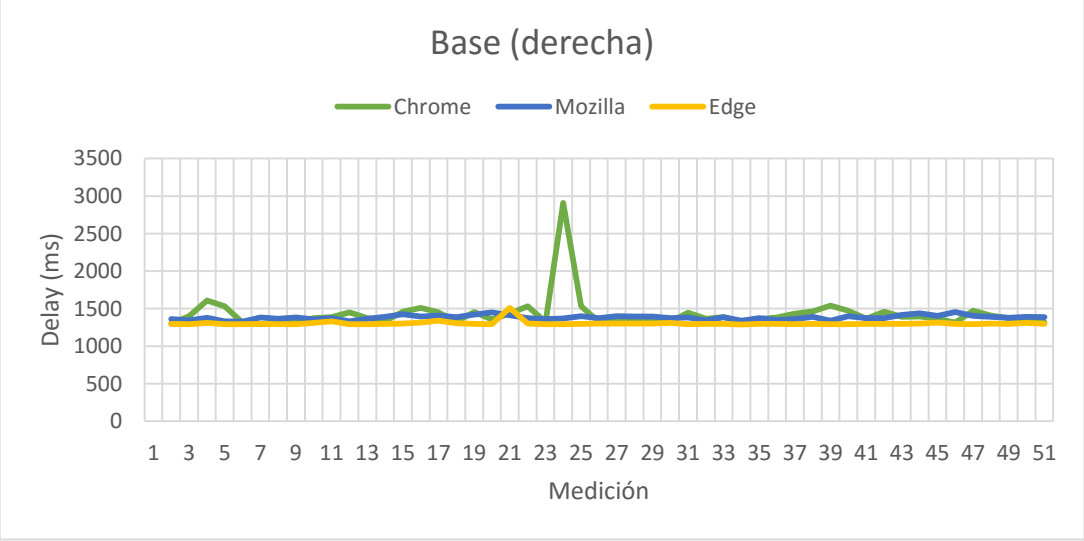


Figura 52: Movimiento de la base hacia la derecha, red LAN, Ethernet v1. Elaboración propia.

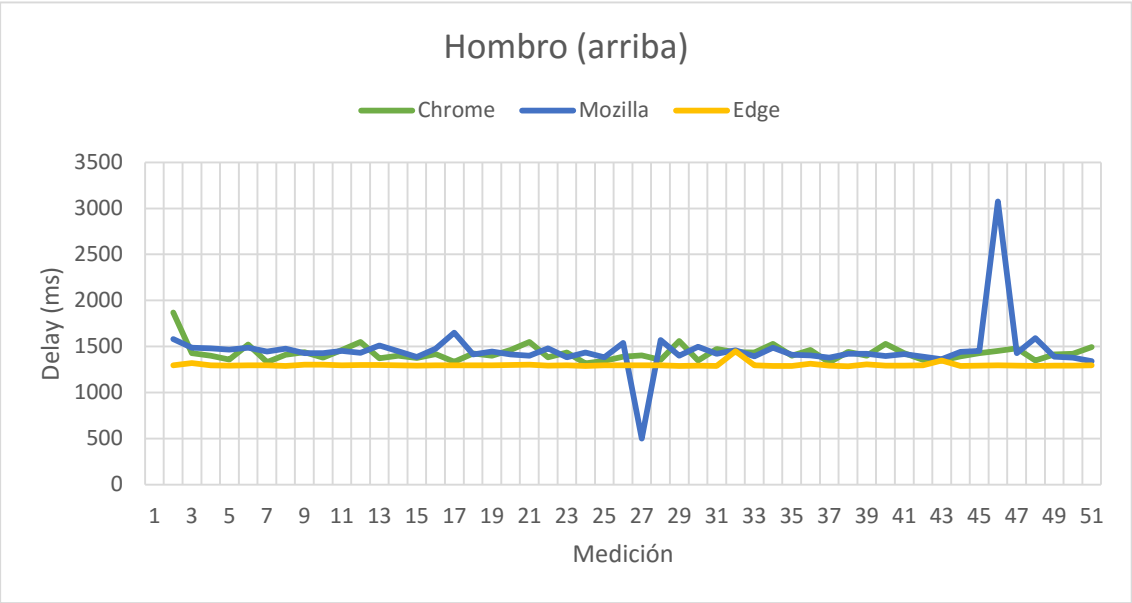


Figura 53: Movimiento del hombro hacia arriba, red LAN, Ethernet v1. Elaboración propia.

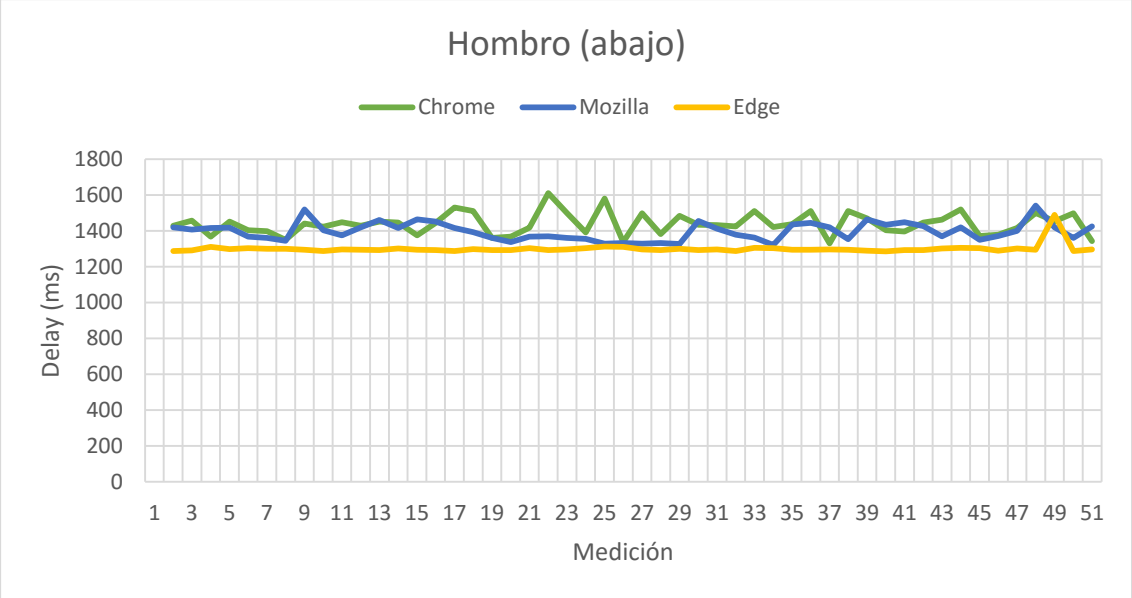


Figura 54: Movimiento del hombro hacia abajo, red LAN, Ethernet v1. Elaboración propia.

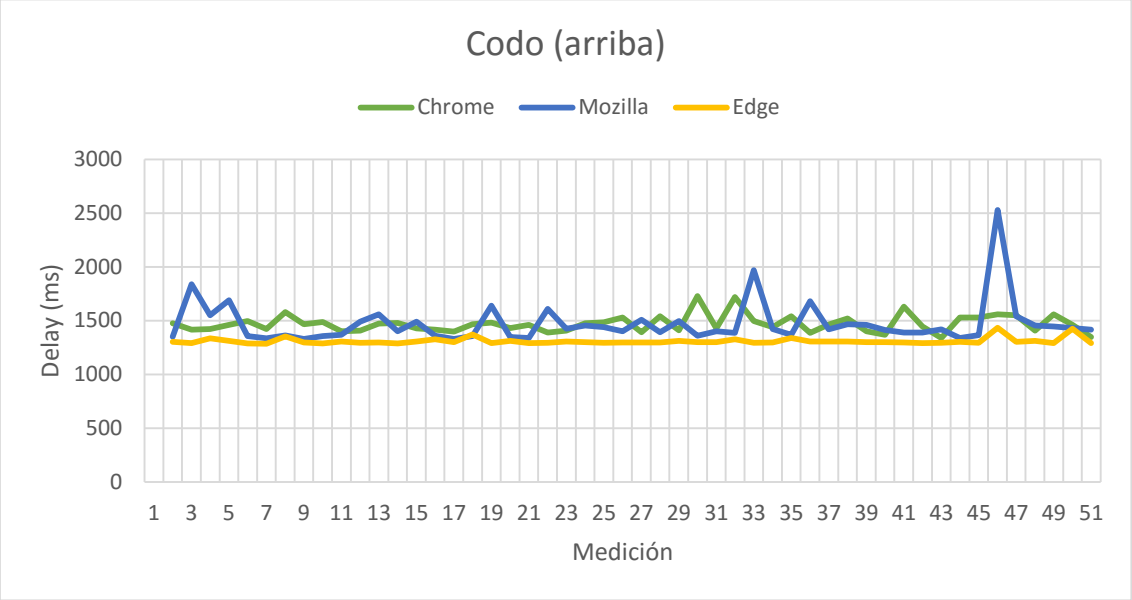


Figura 55: Movimiento del codo hacia arriba, red LAN, Ethernet v1. Elaboración propia.

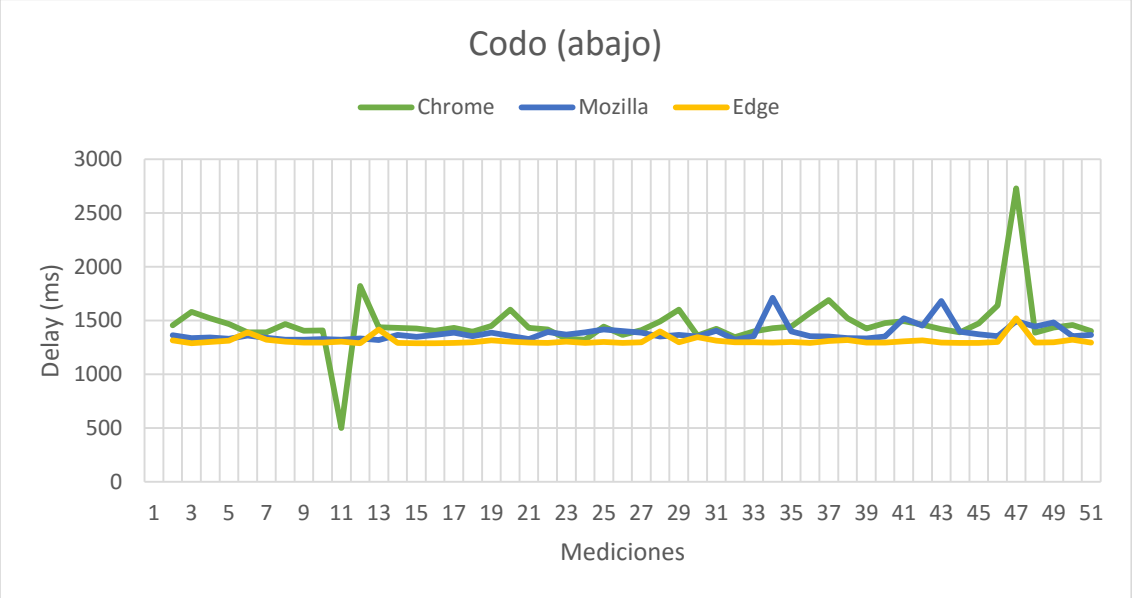


Figura 56: Movimiento del codo hacia abajo, red LAN, Ethernet v1. Elaboración propia.

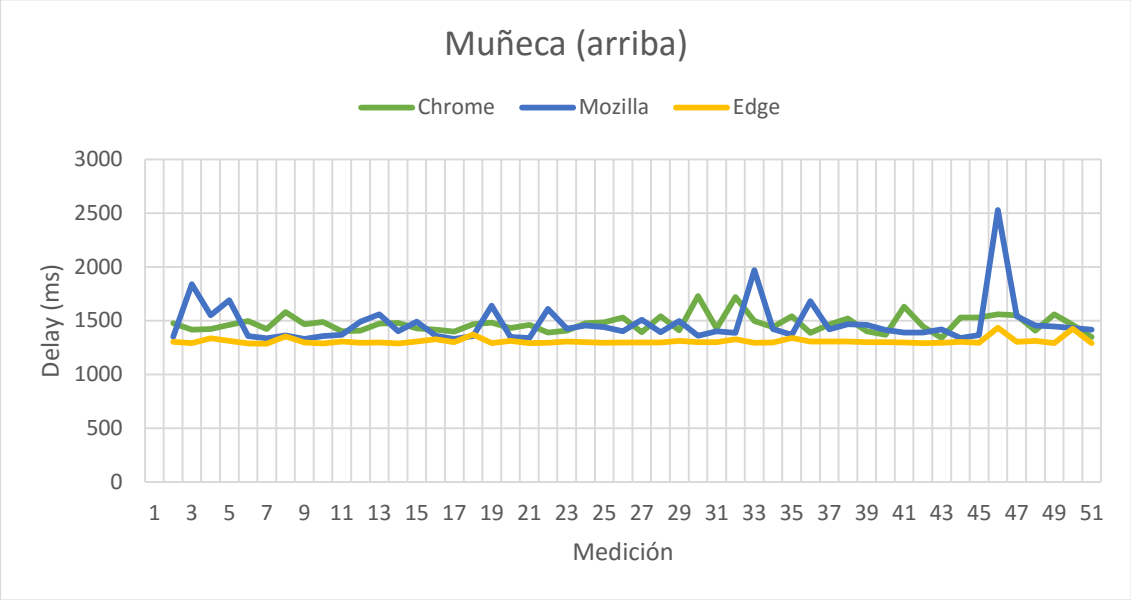


Figura 57: Movimiento de la muñeca hacia arriba, red LAN, Ethernet v1. Elaboración propia.

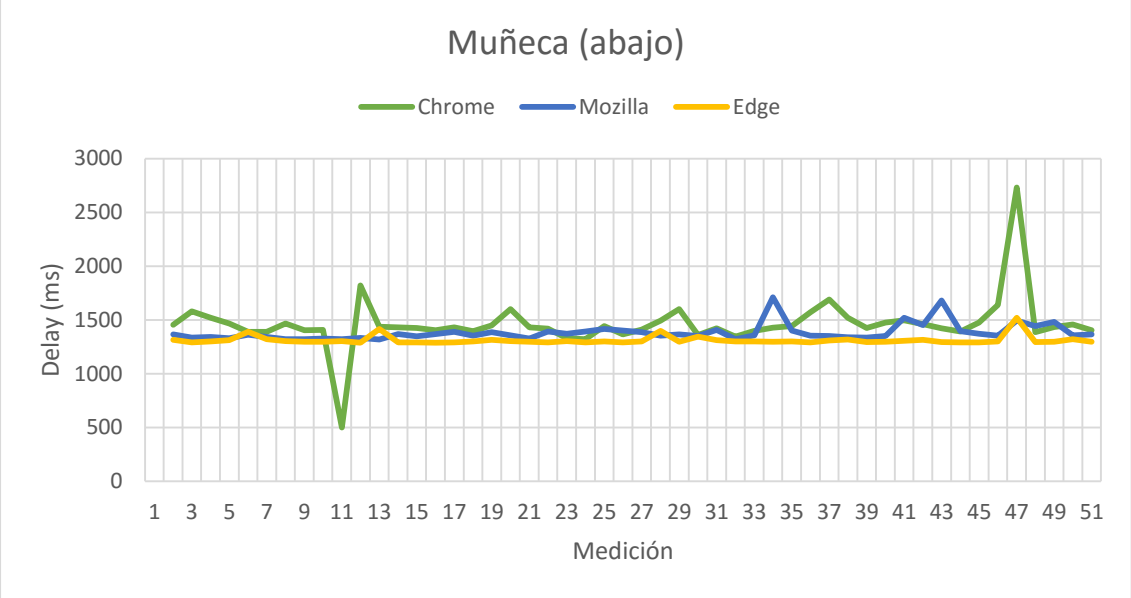


Figura 58: Movimiento de la muñeca hacia arriba, red LAN, Ethernet v1. Elaboración propia.

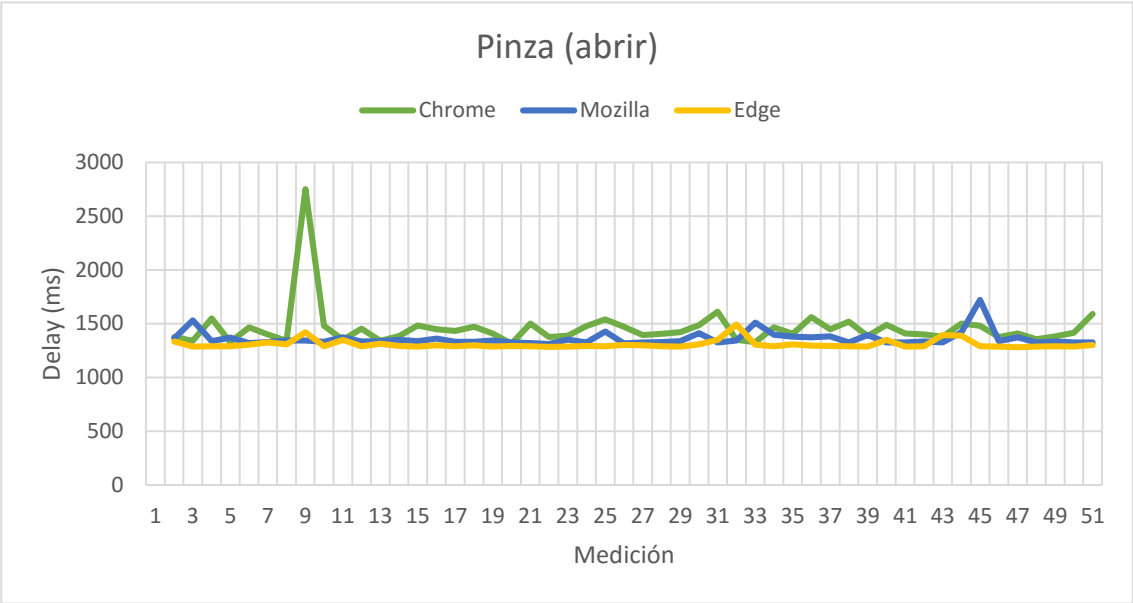


Figura 59: Movimiento al abrir la pinza, red LAN, Ethernet v1. Elaboración propia.

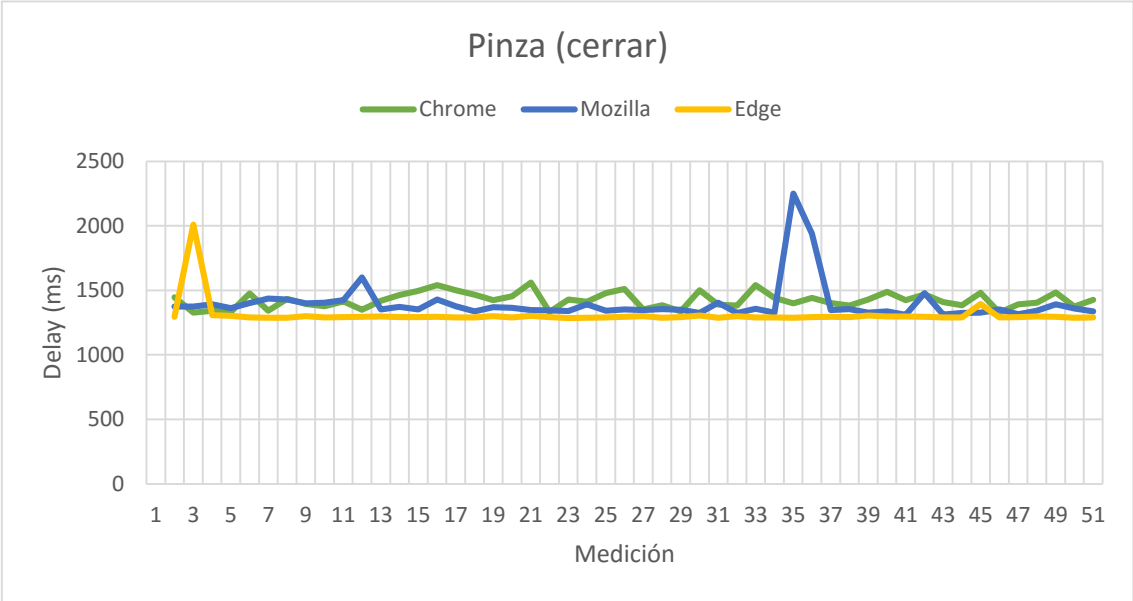


Figura 60: Movimiento al cerrar la pinza, red LAN, Ethernet v1. Elaboración propia.

4. 1. 2. Ethernet v2.

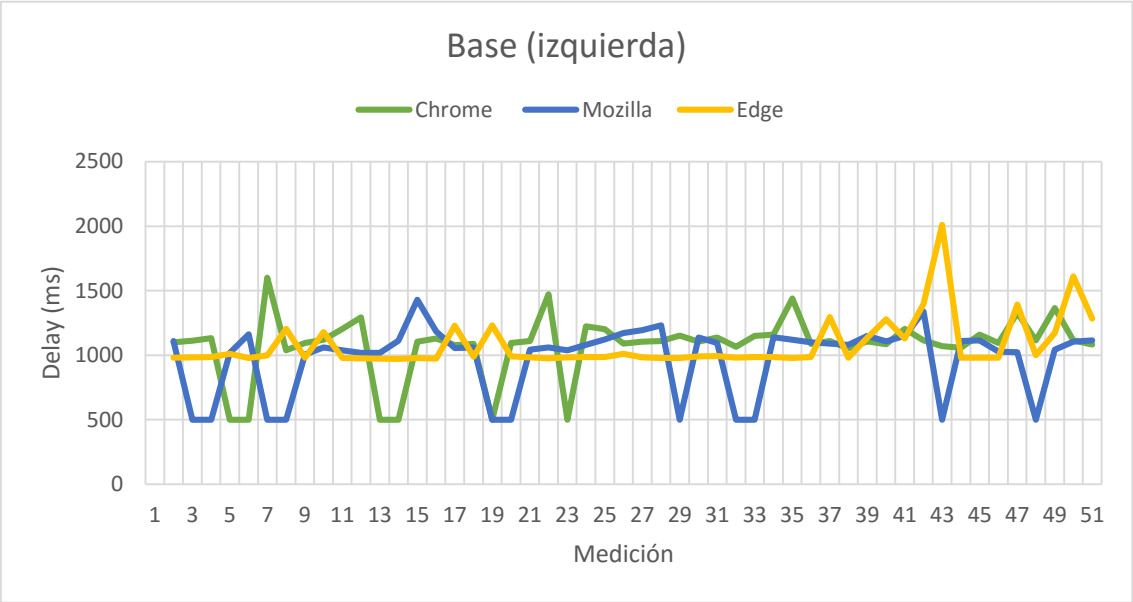


Figura 61: Movimiento de la base a la izquierda, red LAN, Ethernet v2. Elaboración propia.

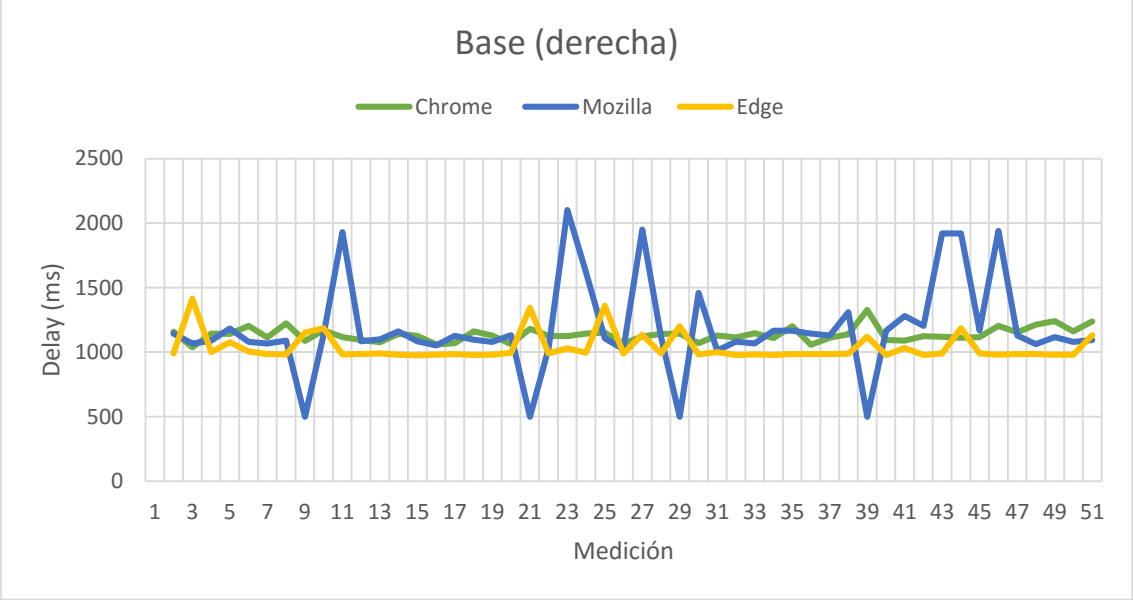


Figura 62: Movimiento de la base a la derecha, red LAN, Ethernet v2. Elaboración propia.

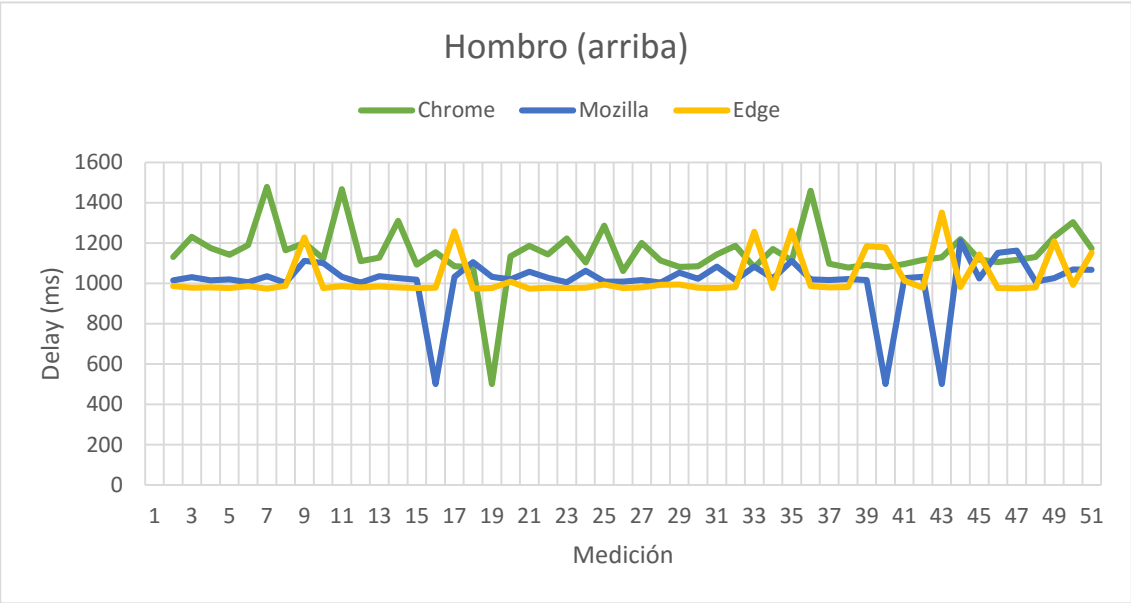


Figura 63: Movimiento del hombro hacia arriba, red LAN, Ethernet v2. Elaboración propia.

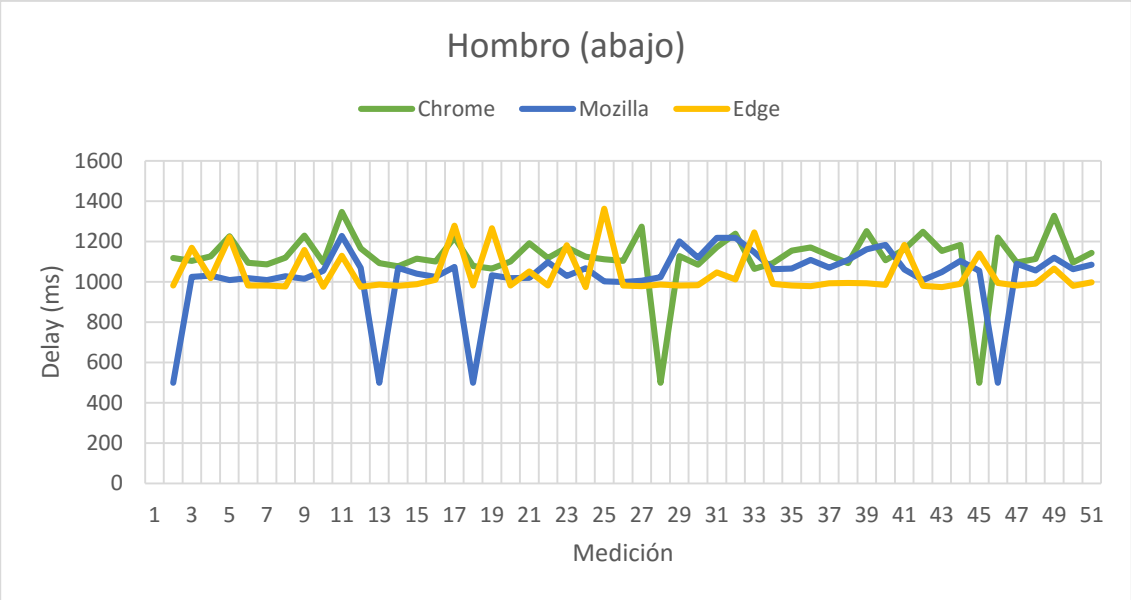


Figura 64: Movimiento del hombro hacia abajo, red LAN, Ethernet v2. Elaboración propia.

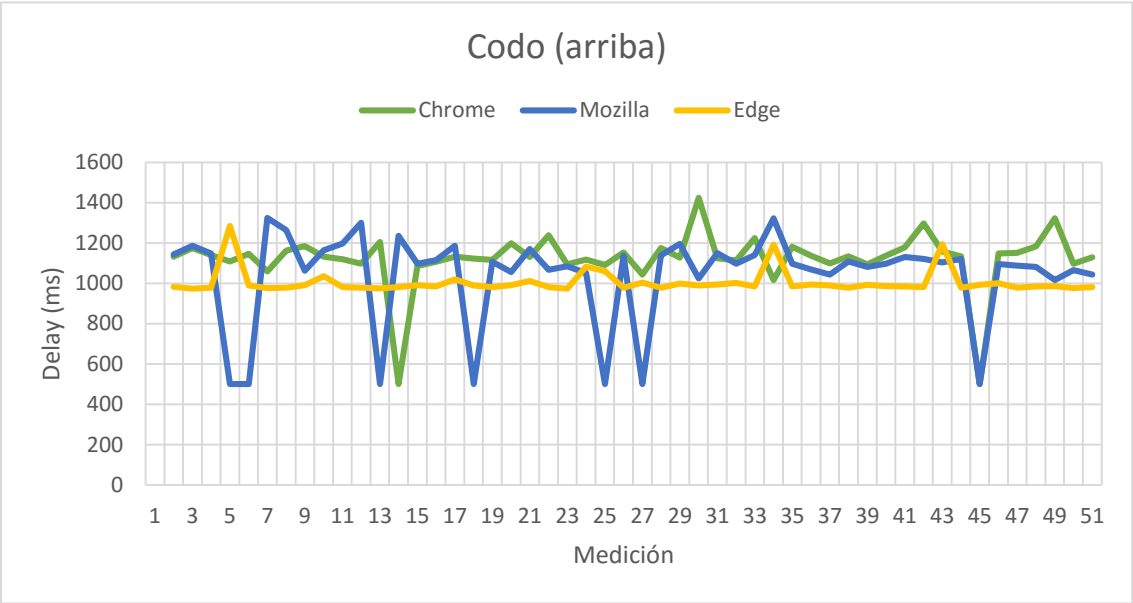


Figura 65: Movimiento del codo hacia arriba, red LAN, Ethernet v2. Elaboración propia.

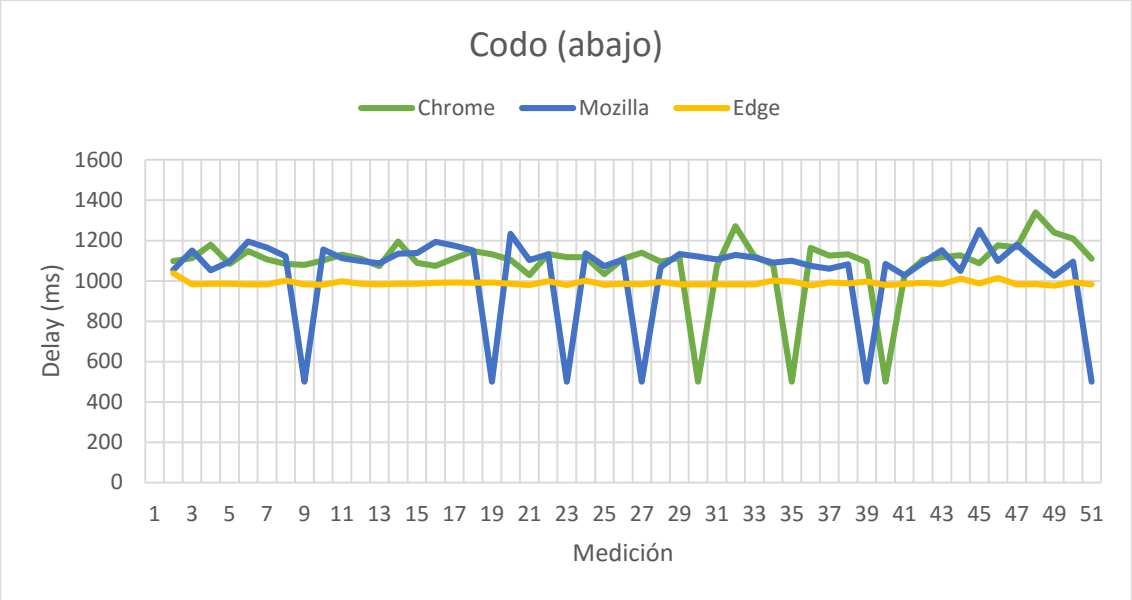


Figura 66: Movimiento del codo hacia abajo, red LAN, Ethernet v2. Elaboración propia.

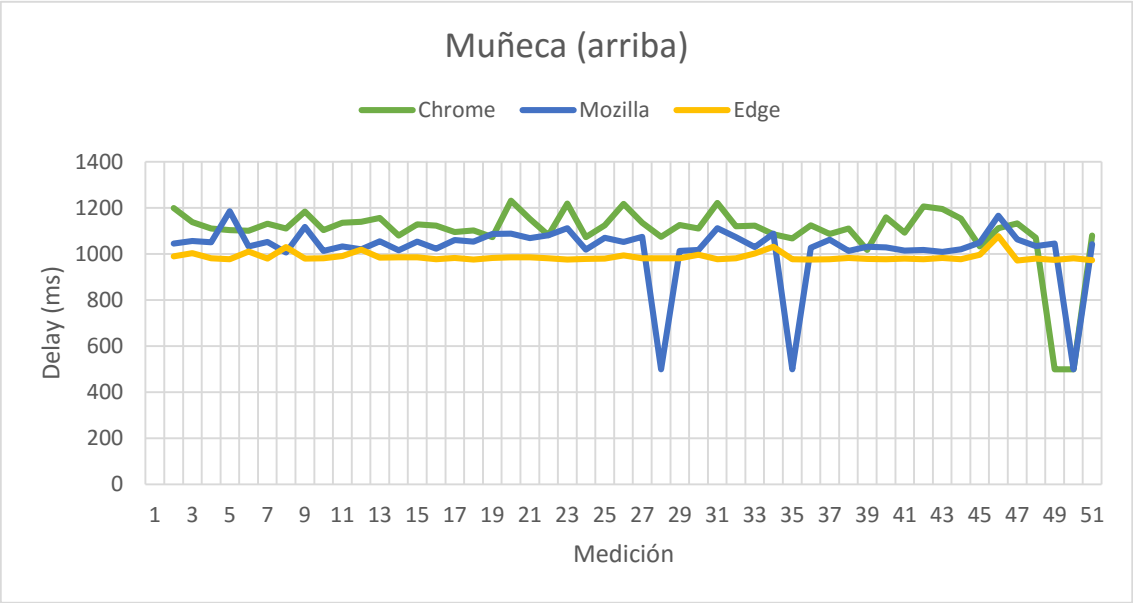


Figura 67: Movimiento de la muñeca hacia arriba, red LAN, Ethernet v2. Elaboración propia.

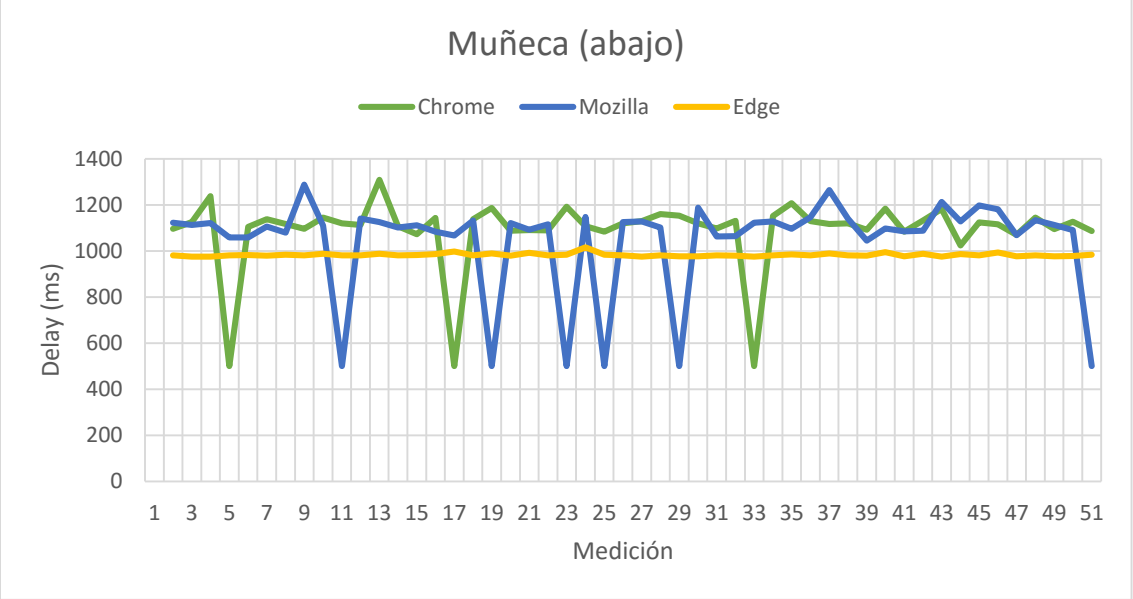


Figura 68: Movimiento de la muñeca hacia abajo, red LAN, Ethernet v2. Elaboración propia.

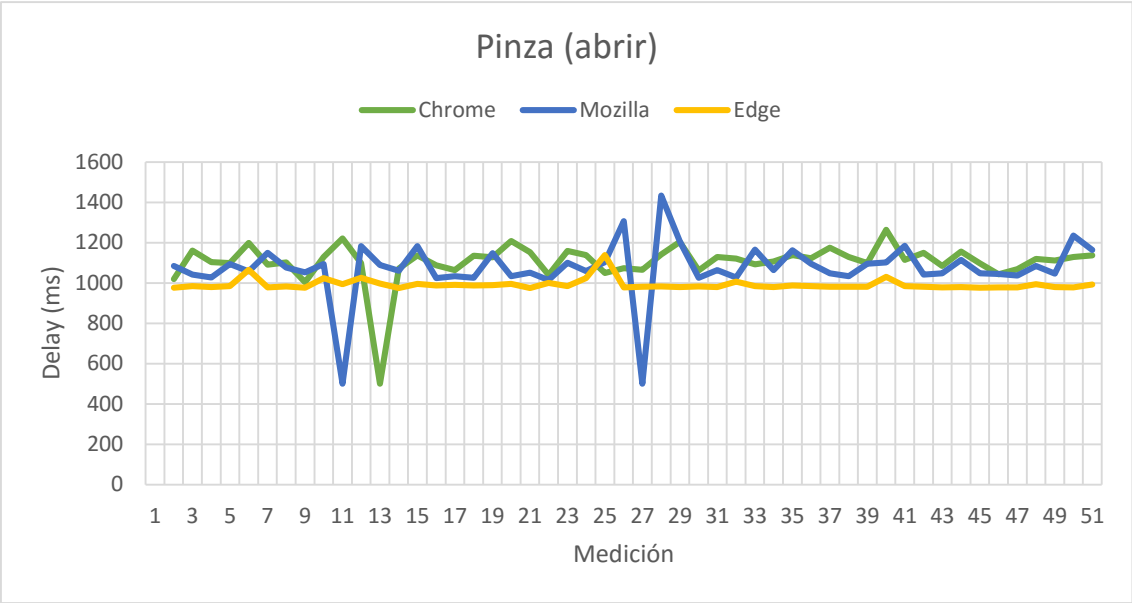


Figura 69: Movimiento al cerrar la pinza, red LAN, Ethernet v2. Elaboración propia.

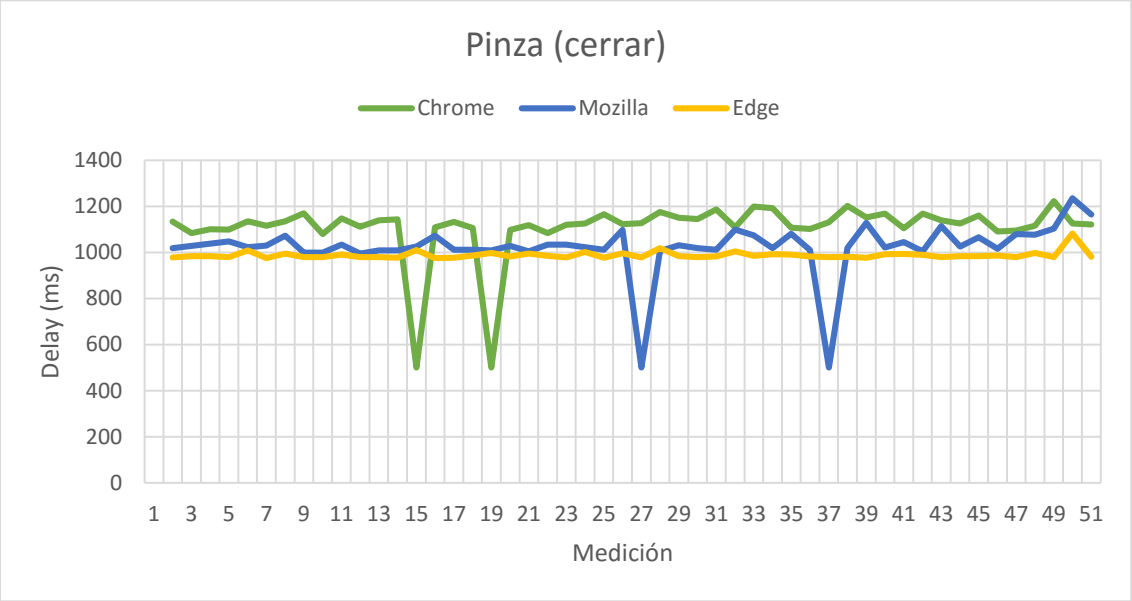


Figura 70: Movimiento al cerrar la pinza, red LAN, Ethernet v2. Elaboración propia.

4. 1. 3. Wi-Fi v1.

Al W5100 le fue asignada una dirección IP de manera manual en el código de programación, sin embargo, también existe la posibilidad de que dicha dirección IP sea asignada automáticamente, este método de asignación fue utilizado en el NodeMCU con el objetivo de ejemplificar su funcionamiento.

Para realizar lo anterior se utiliza la instrucción ***IPAddress myIP = WiFi.softAPIP()*** dentro de la función void setup ()***,*** con la cual se seleccionará una dirección IP de entre las disponibles en la red.

Aunque podría parecer un método más rápido y sencillo tiene el inconveniente de que no sabremos qué dirección IP fue asignada o si la asignación fallo por lo que para conocer dicha dirección podemos mostrarla en el monitor serie de Arduino o bien podemos verificarla directamente en la configuración de nuestro router.

Una vez aclarado lo anterior, procederemos a mostrar los resultados de las pruebas realizadas.

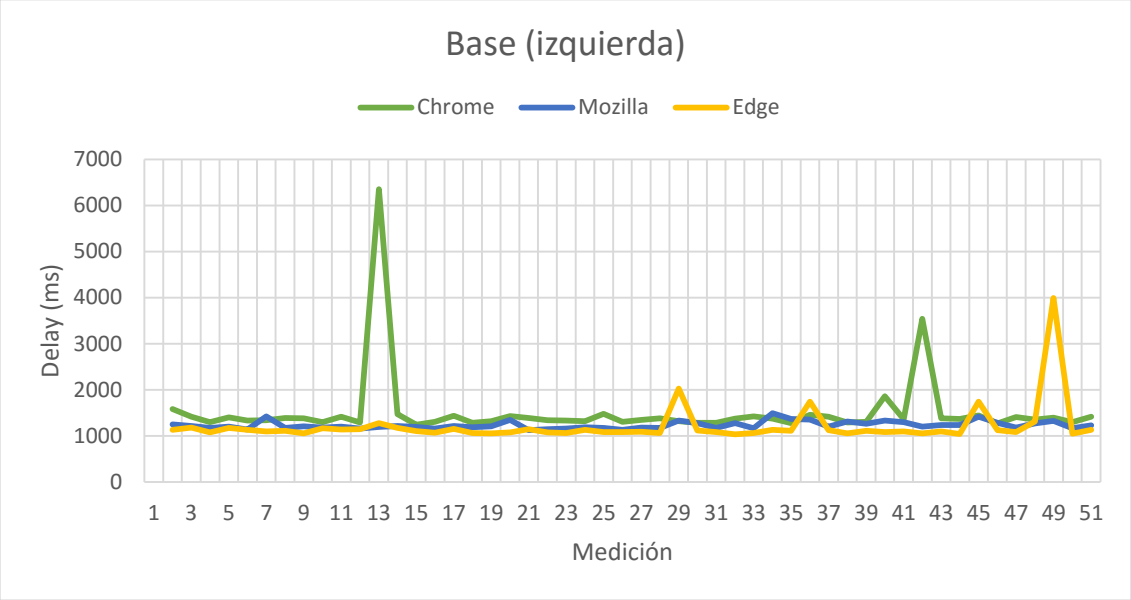


Figura 71: Movimiento de la base a la izquierda, red LAN, Wi-Fi v1. Elaboración propia.

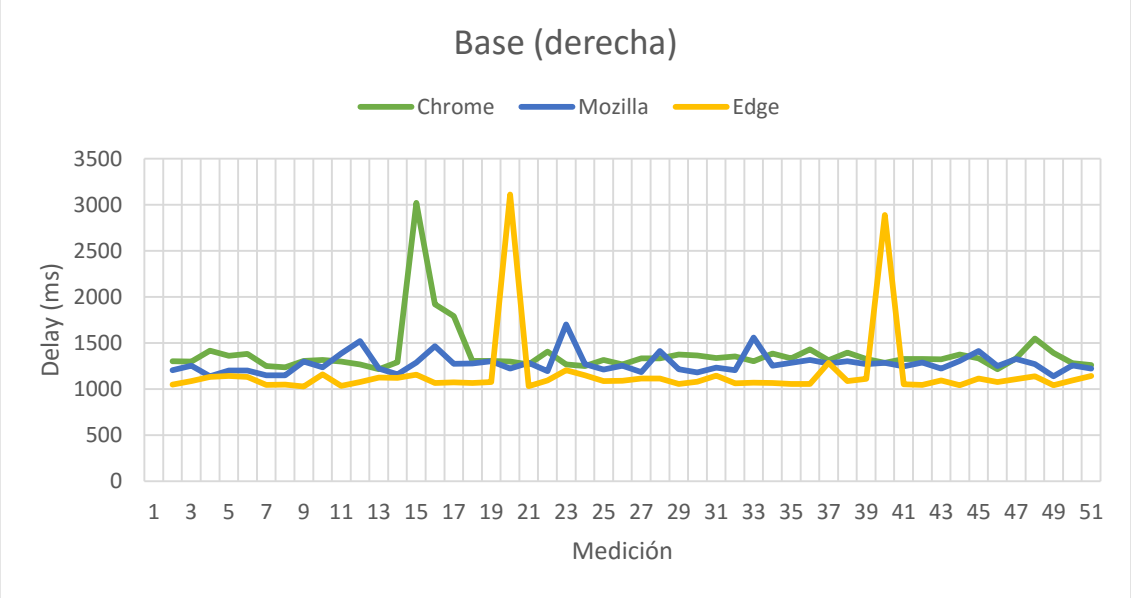


Figura 72: Movimiento de la base a la derecha, red LAN, Wi-Fi v1. Elaboración propia.

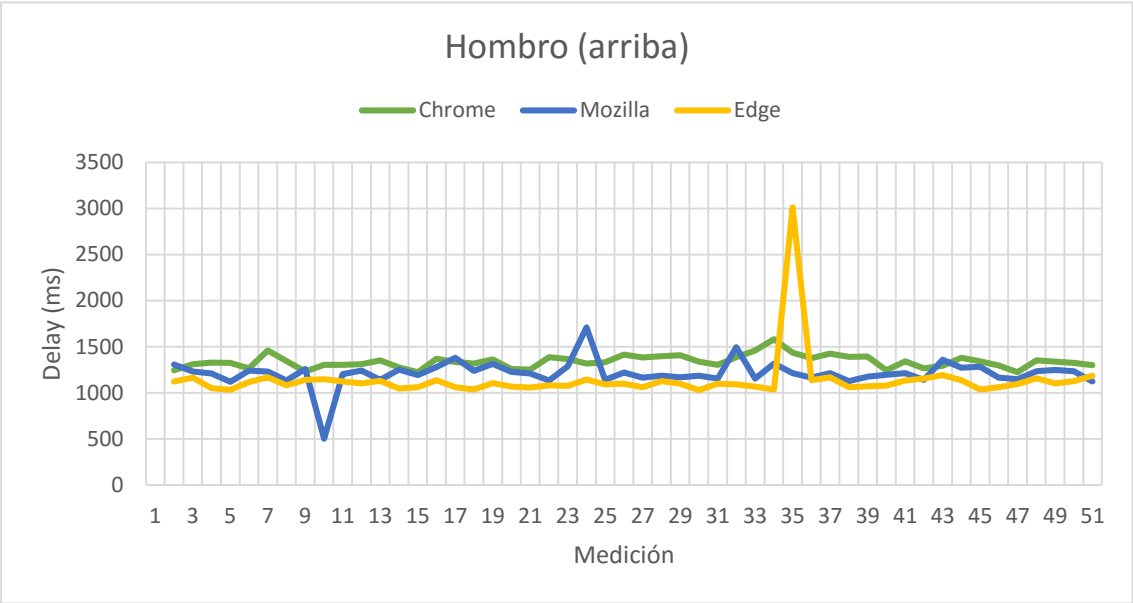


Figura 73: Movimiento del hombro hacia arriba, red LAN, Wi-Fi v1. Elaboración propia.

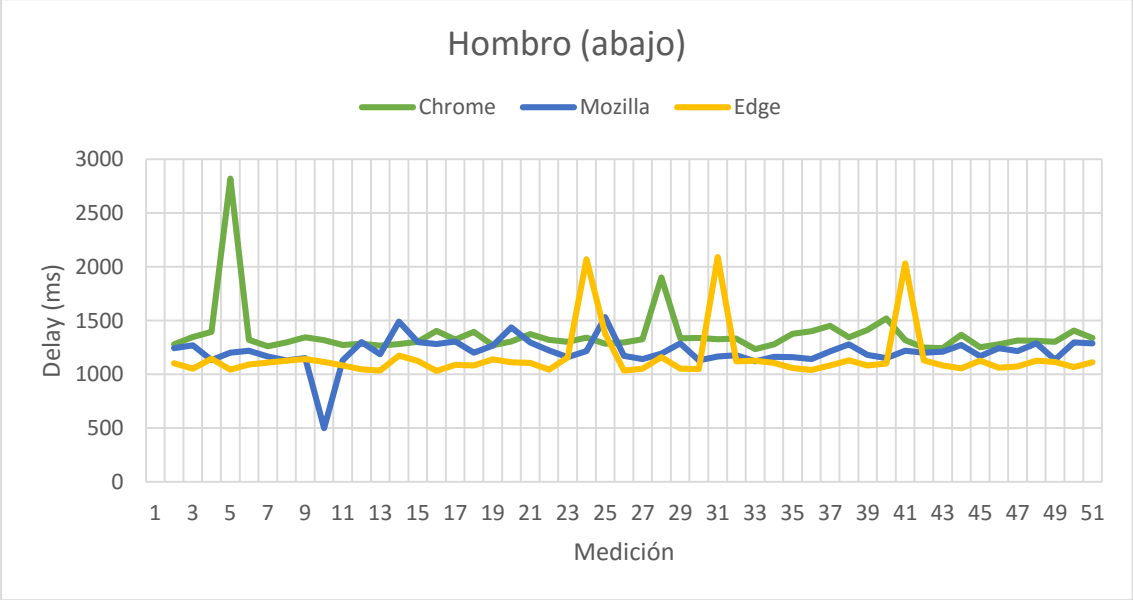


Figura 74: Movimiento del hombro hacia abajo, red LAN, Wi-Fi v1. Elaboración propia.

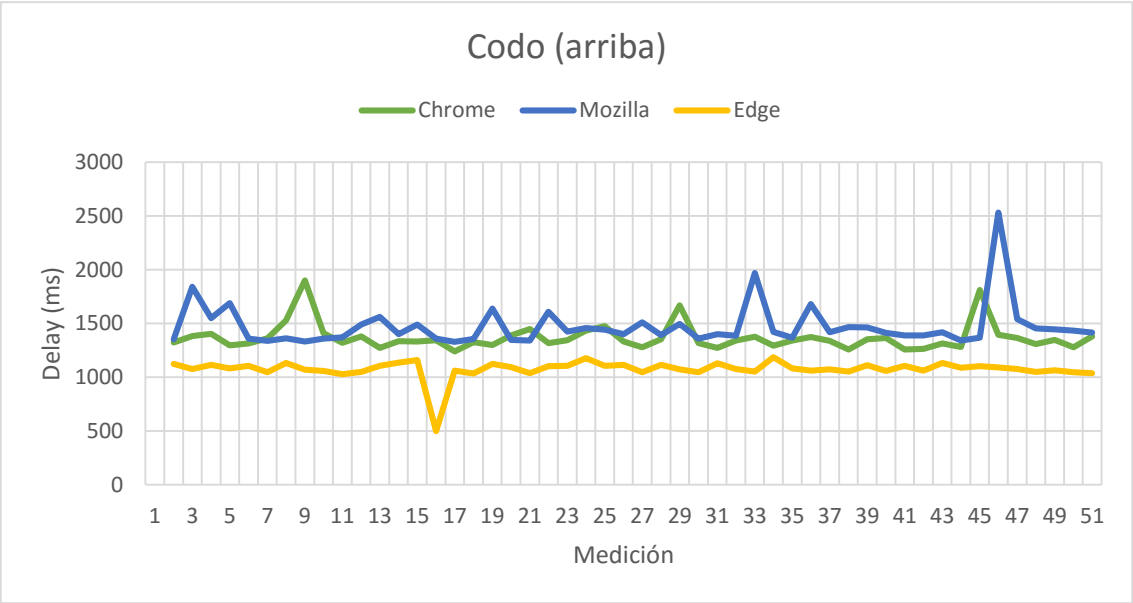


Figura 75: Movimiento del codo hacia arriba, red LAN, Wi-Fi v1. Elaboración propia.

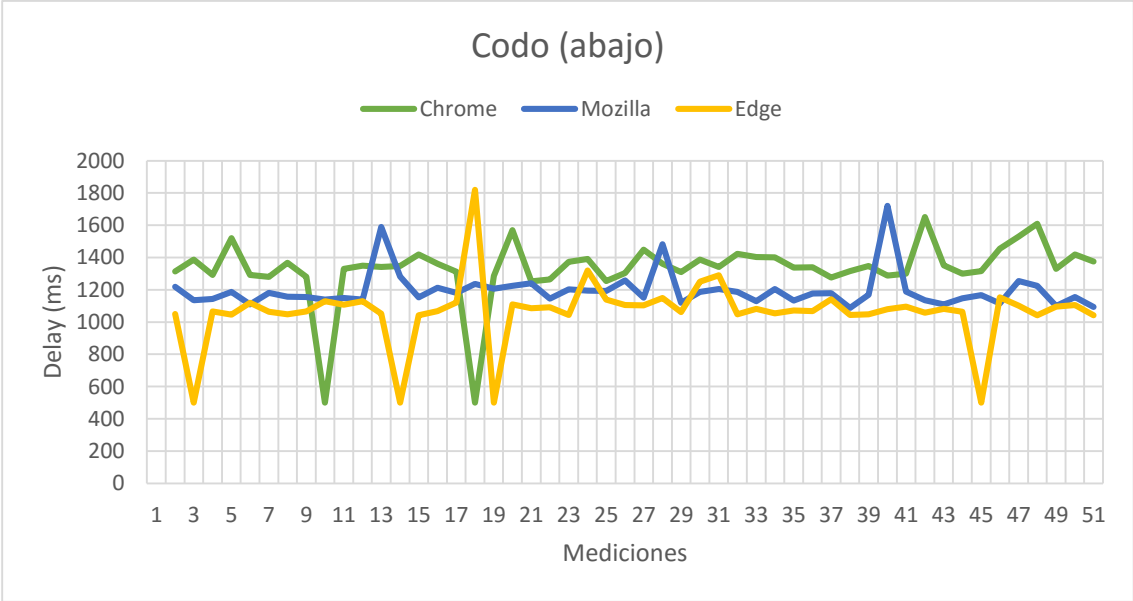


Figura 76: Movimiento del codo hacia abajo, red LAN, Wi-Fi v1. Elaboración propia.

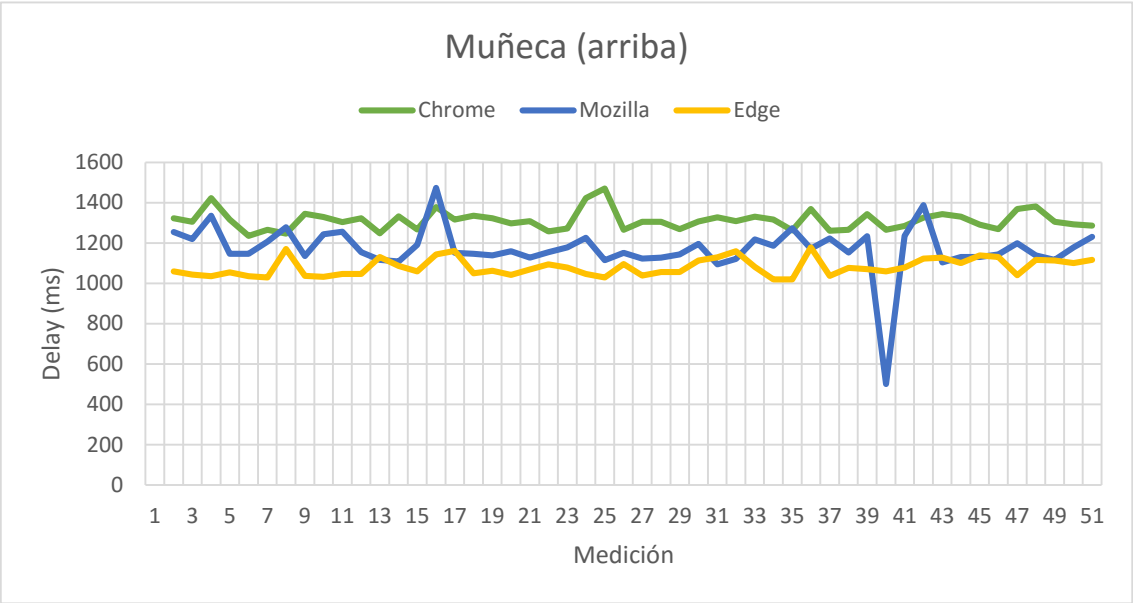


Figura 78: Movimiento de la muñeca hacia arriba, red LAN, Wi-Fi v1. Elaboración propia.

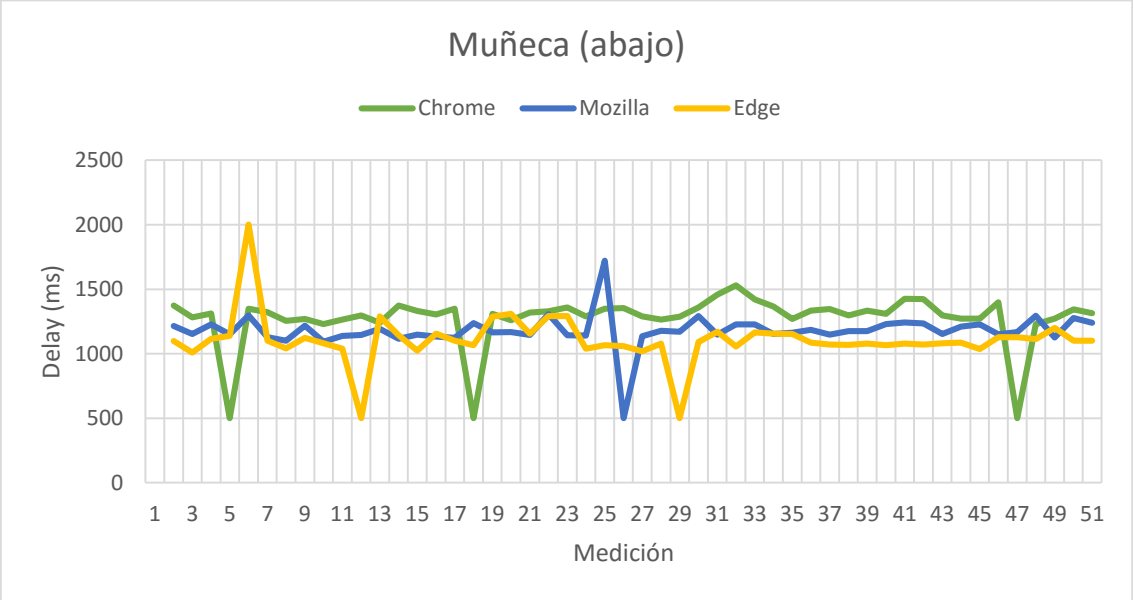


Figura 77: Movimiento de la muñeca hacia abajo, red LAN, Wi-Fi v1. Elaboración propia.

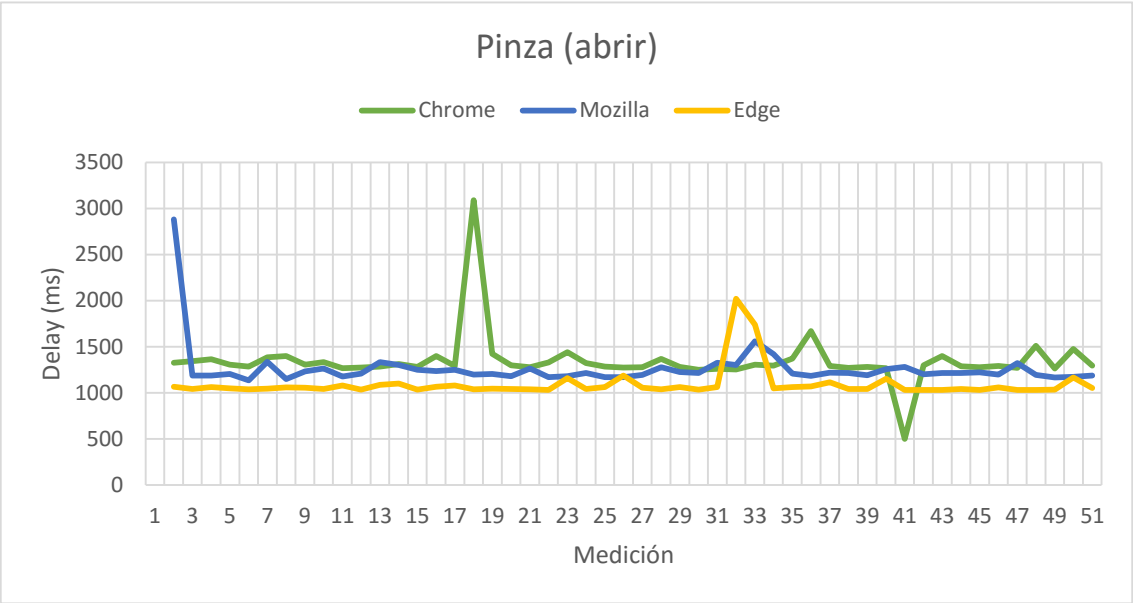


Figura 79: Movimiento al abrir la pinza, red LAN, Wi-Fi v1. Elaboración propia.

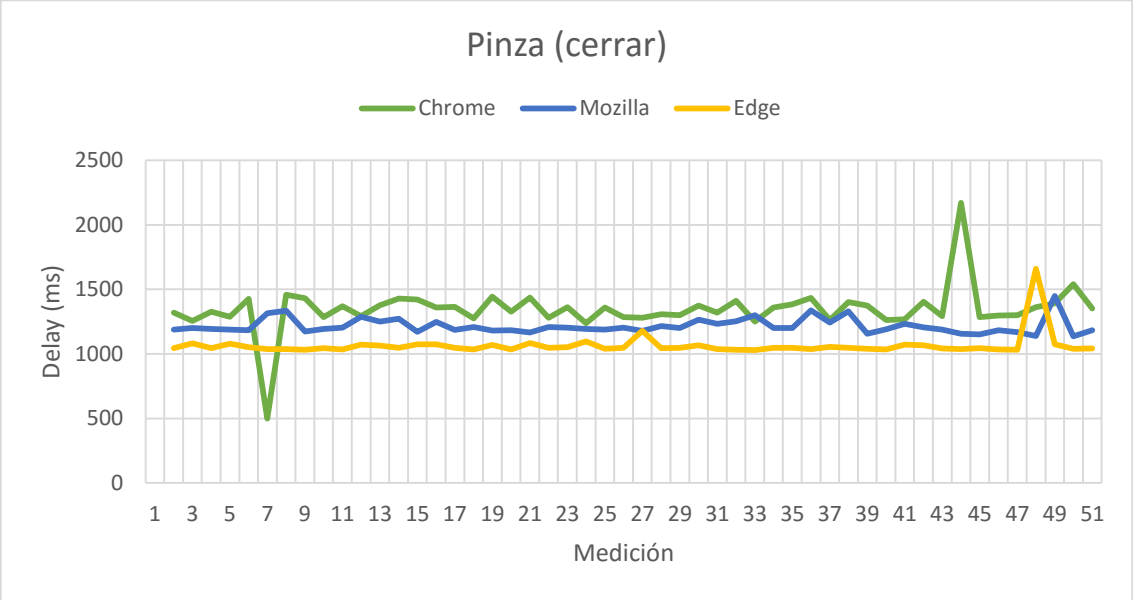


Figura 80: Movimiento al cerrar la pinza, red LAN, Wi-Fi v1. Elaboración propia.

4. 1. 4 Wi-Fi v2.

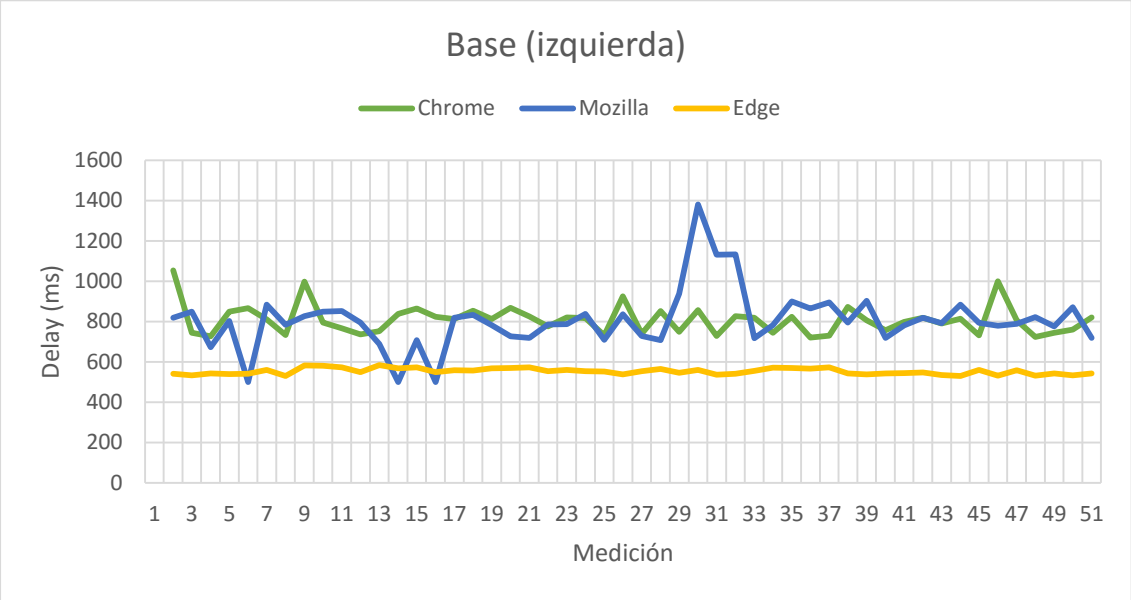


Figura 81: Movimiento de la base a la izquierda, red LAN, Wi-Fi v2. Elaboración propia.

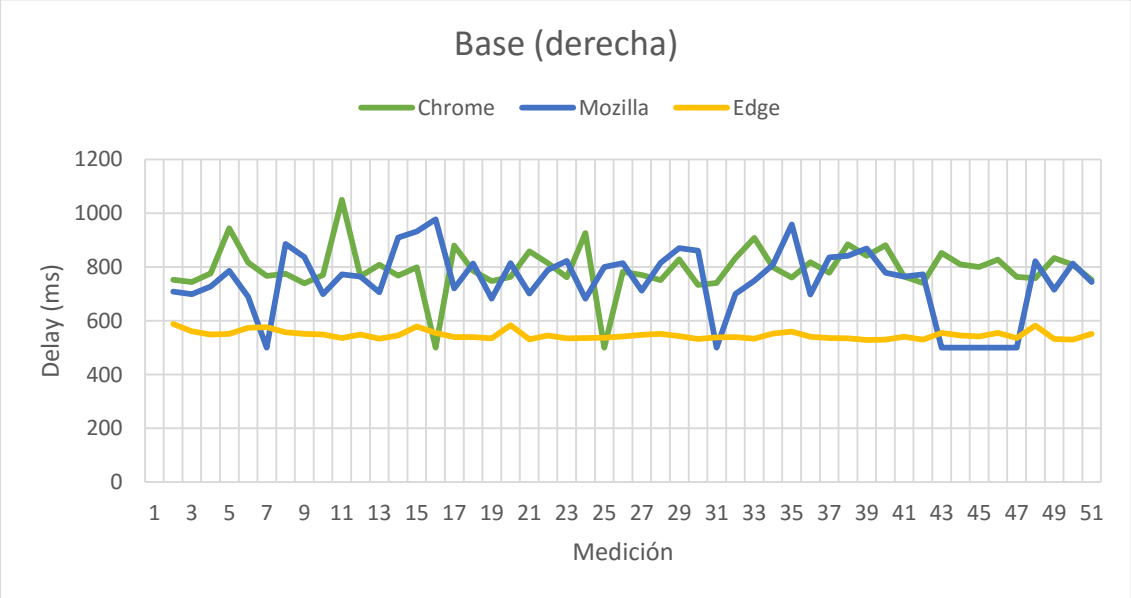


Figura 82: Movimiento de la base a la derecha, red LAN, Wi-Fi v2. Elaboración propia.

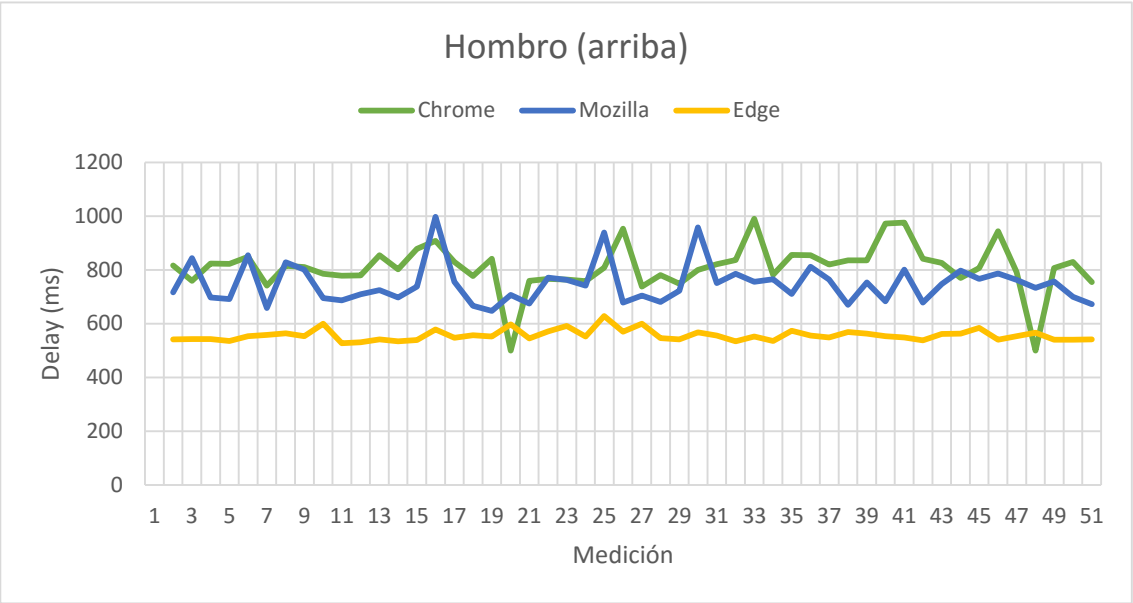


Figura 83: Movimiento del hombro hacia arriba, red LAN, Wi-Fi v2. Elaboración propia.

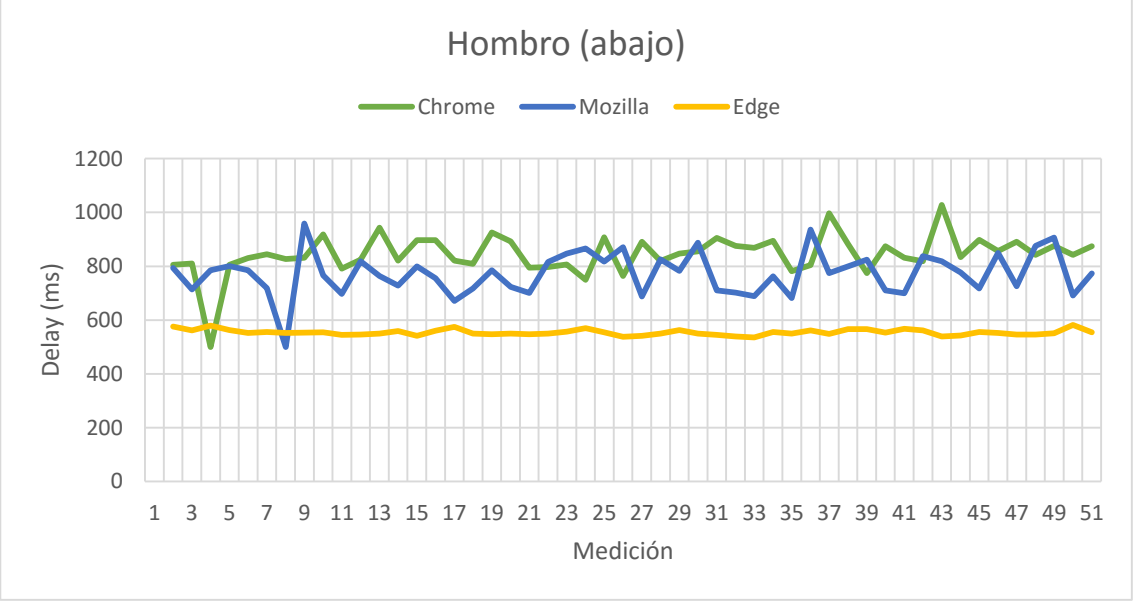


Figura 84: Movimiento del hombro hacia abajo, red LAN, Wi-Fi v2. Elaboración propia.

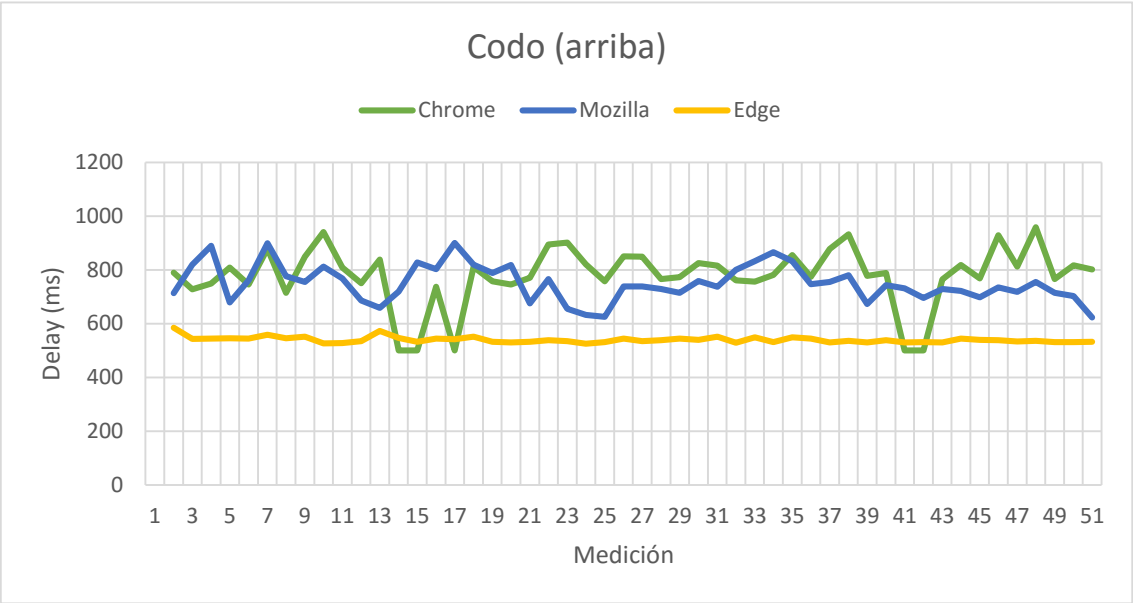


Figura 85: Movimiento del codo hacia arriba, red LAN, Wi-Fi v2. Elaboración propia.

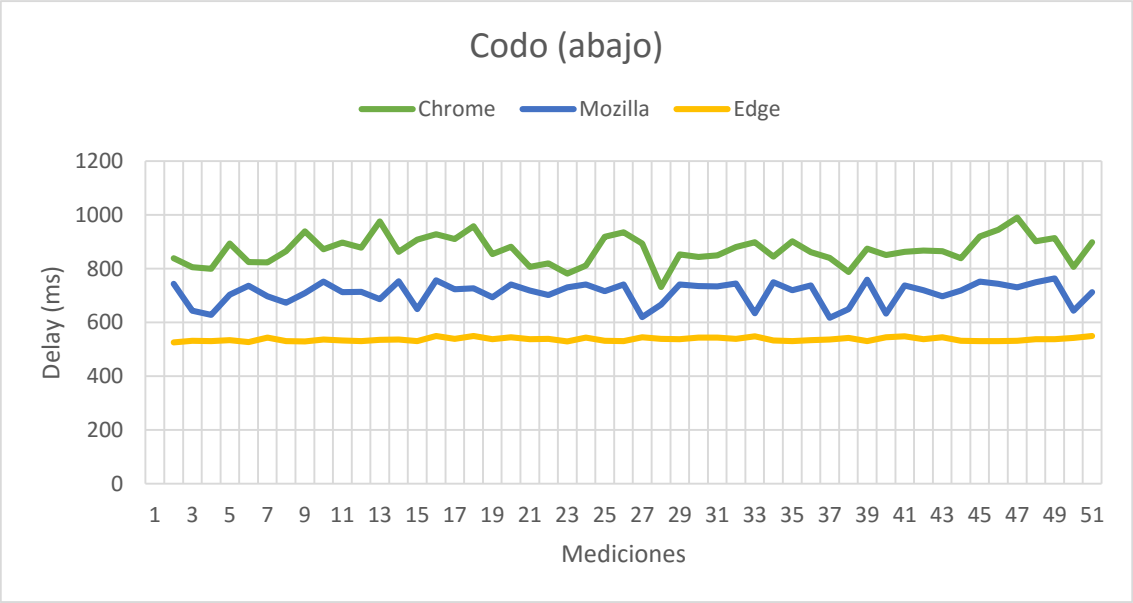


Figura 86: Movimiento del codo hacia abajo, red LAN, Wi-Fi v2. Elaboración propia.

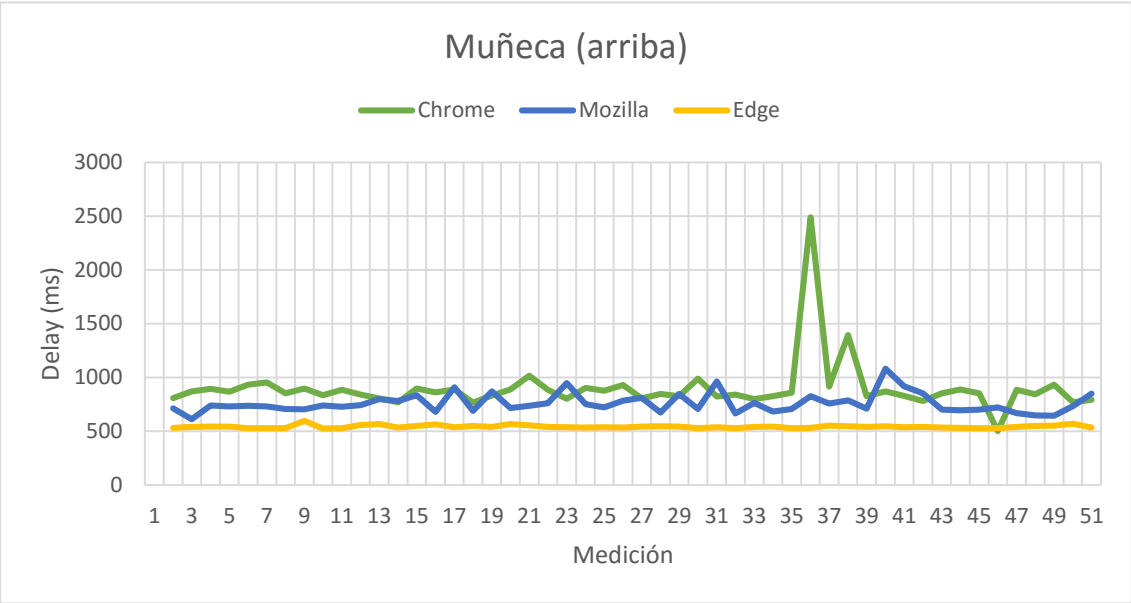


Figura 87: Movimiento de la muñeca hacia arriba, red LAN, Wi-Fi v2. Elaboración propia.

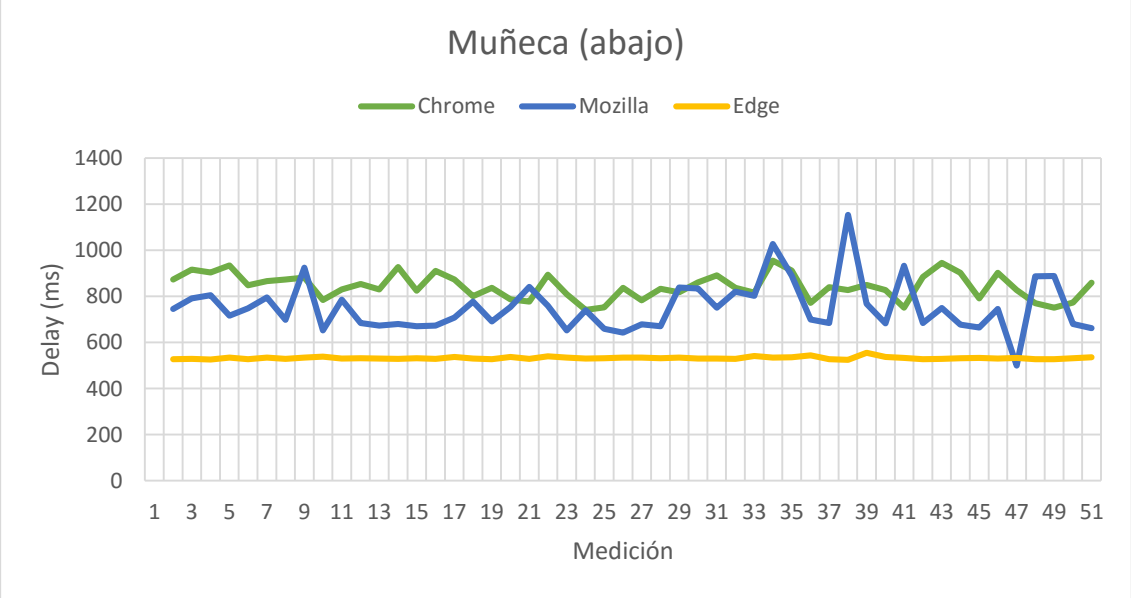


Figura 88: Movimiento de la muñeca hacia abajo, red LAN, Wi-Fi v2. Elaboración propia.

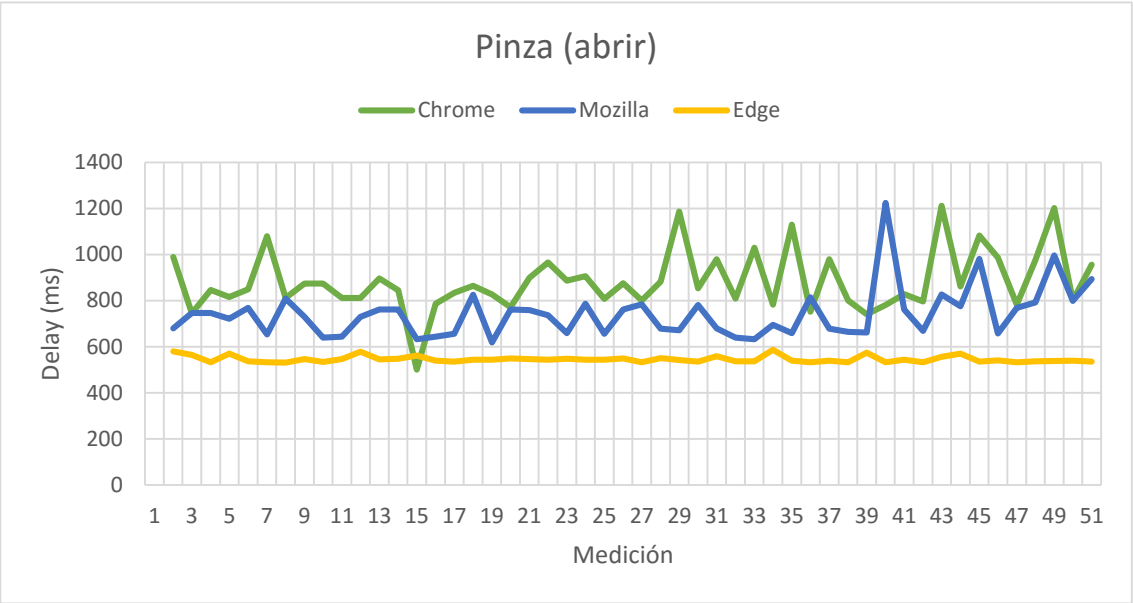


Figura 89: Movimiento al abrir la pinza, red LAN, Wi-Fi v2. Elaboración propia.

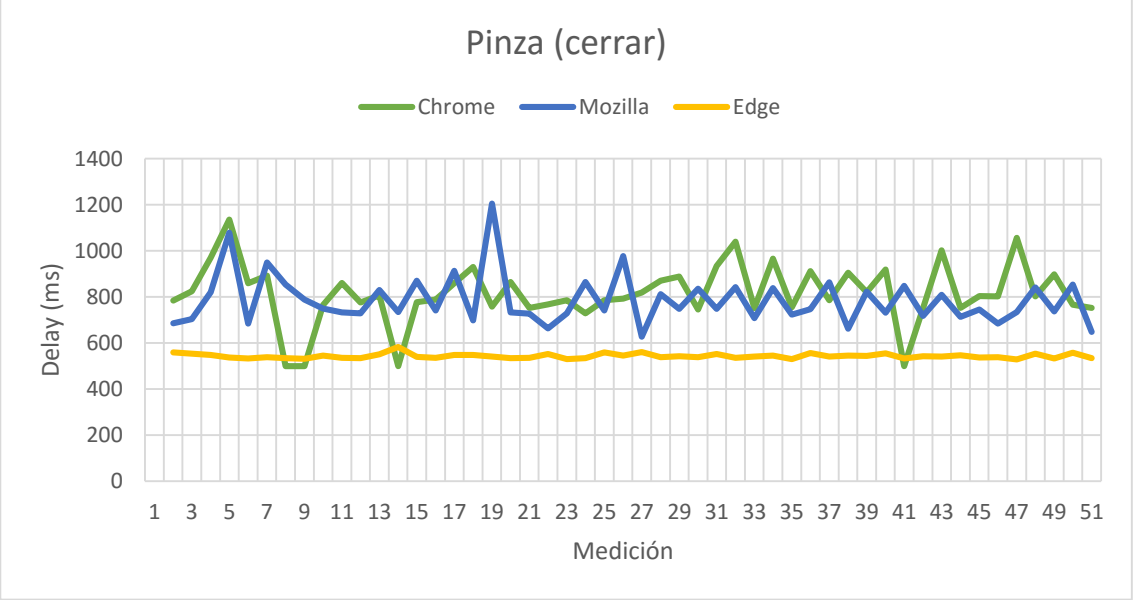


Figura 90: Movimiento al cerrar la pinza, red LAN, Wi-Fi v2. Elaboración propia.

4. 2 Red WAN.

En el caso de las redes LAN la dirección que debe escribirse en el buscador para ingresar a la página de control es la dirección IP que posee el dispositivo, ya sea que fuera asignada por el usuario o que esta fuera asignada automáticamente, sin embargo, para conectarse al servidor desde una red externa se requiere conocer el puerto de conexión (en nuestro caso se eligió el 80 para W5100 y NodeMCU) y la dirección IP pública de nuestro router.

Para conocer esa dirección basta con entrar a la página web ***https://www.whatismyip.com/es/*** y en ella se mostrará nuestra IP pública.

Figura 91: Información mostrada por whatismyip. La IP pública aparece como IPv4 público. Elaboración propia.

En el buscador escribiremos la dirección IP seguida de dos puntos y el puerto de conexión, de la siguiente manera: *187.190.15.27:80*

4. 2. 1. Ethernet v1.

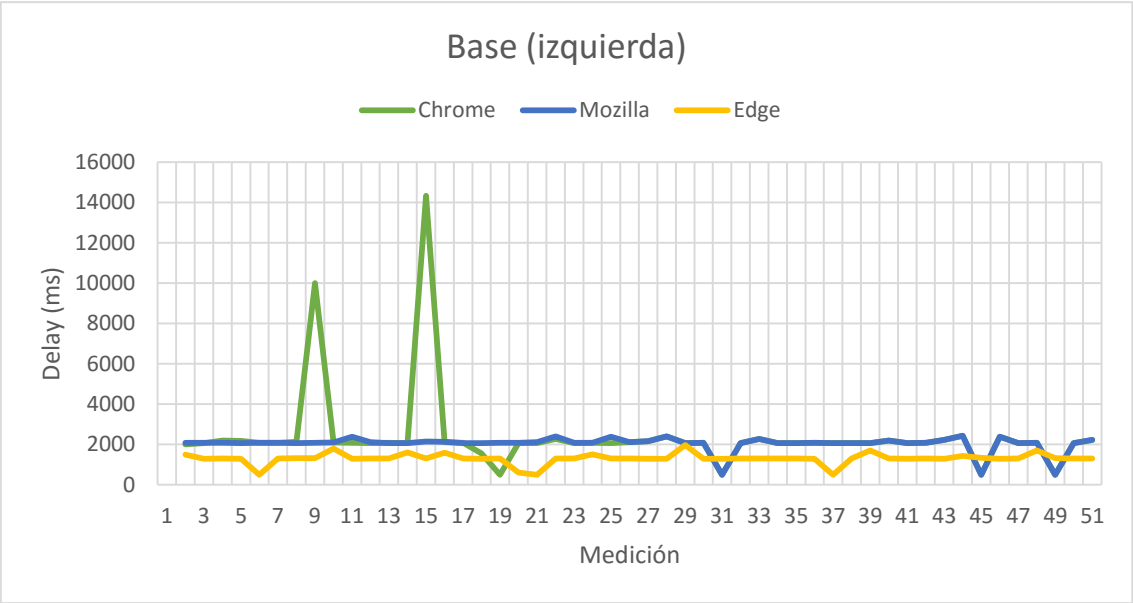


Figura 92: Movimiento de la base a la izquierda, red WAN, Ethernet v1. Elaboración propia.

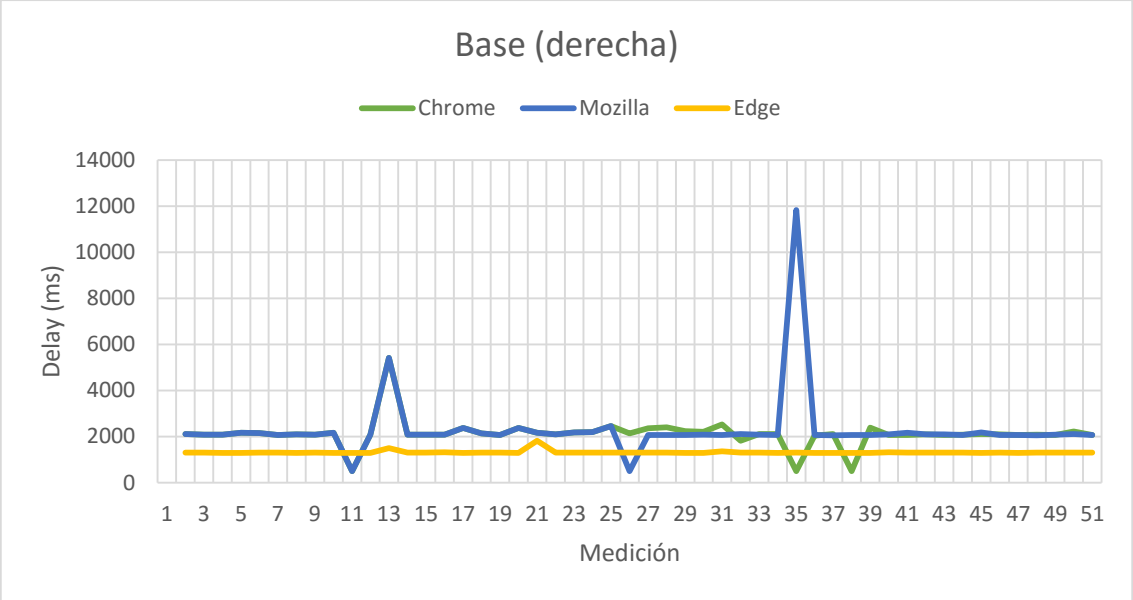


Figura 93: Movimiento de la base a la derecha, red WAN, Ethernet v1. Elaboración propia.

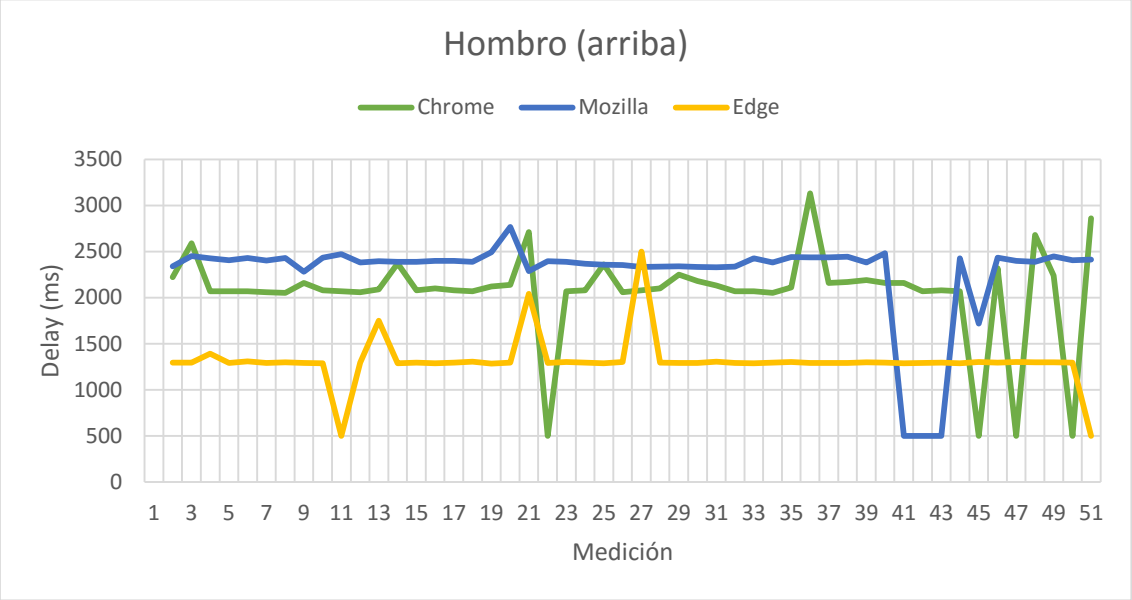


Figura 94: Movimiento del hombro hacia arriba, red WAN, Ethernet v1. Elaboración propia.

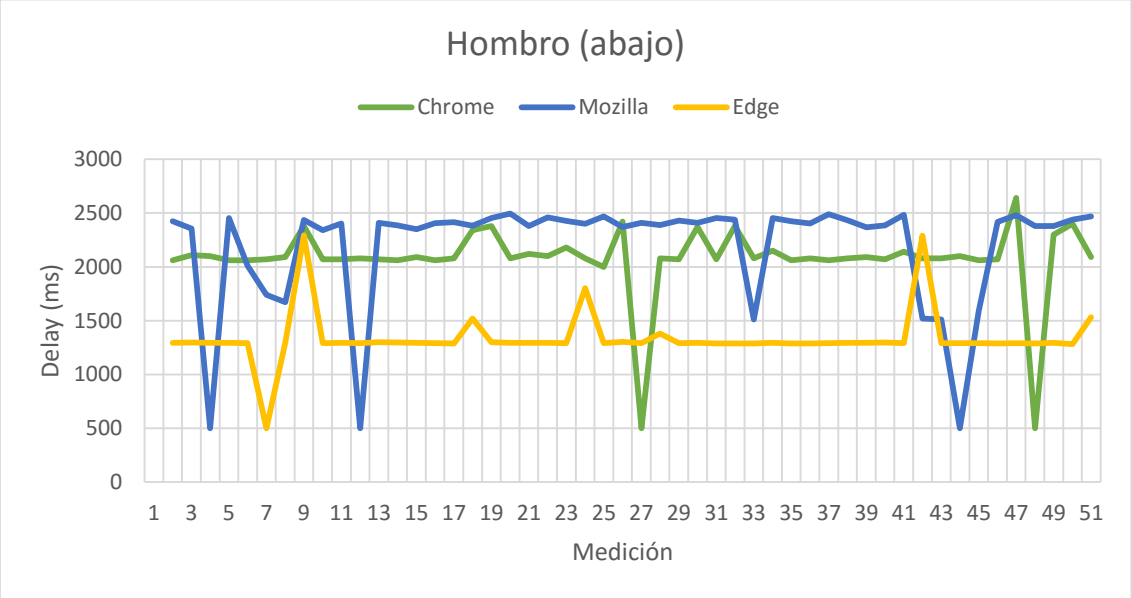


Figura 95: Movimiento del hombro hacia abajo, red WAN, Ethernet v1. Elaboración propia.

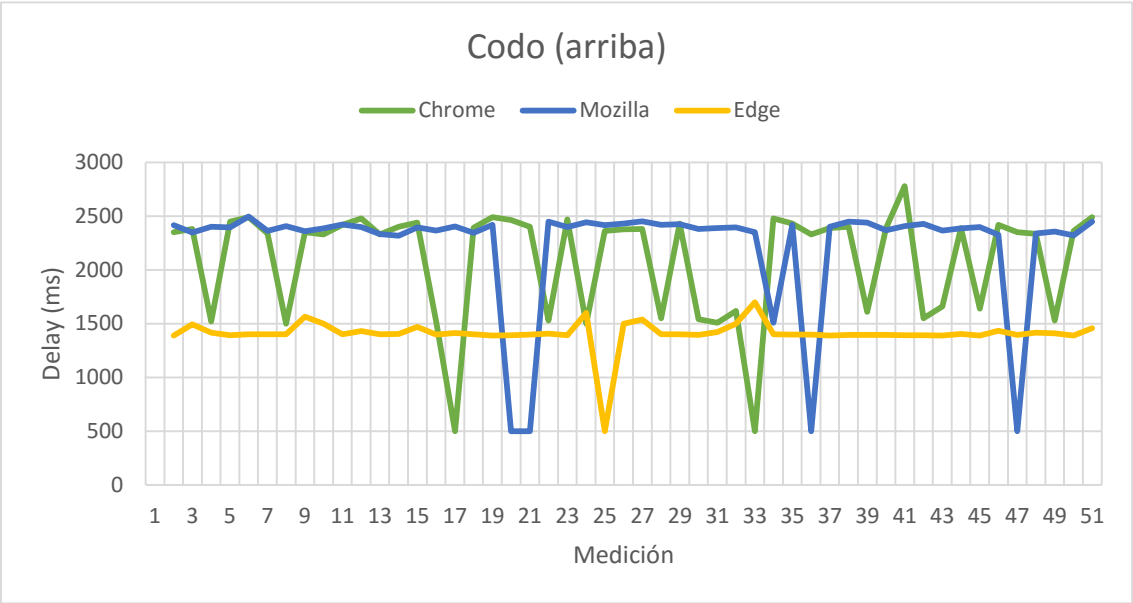


Figura 96: Movimiento del codo hacia arriba, red WAN, Ethernet v1. Elaboración propia.

x

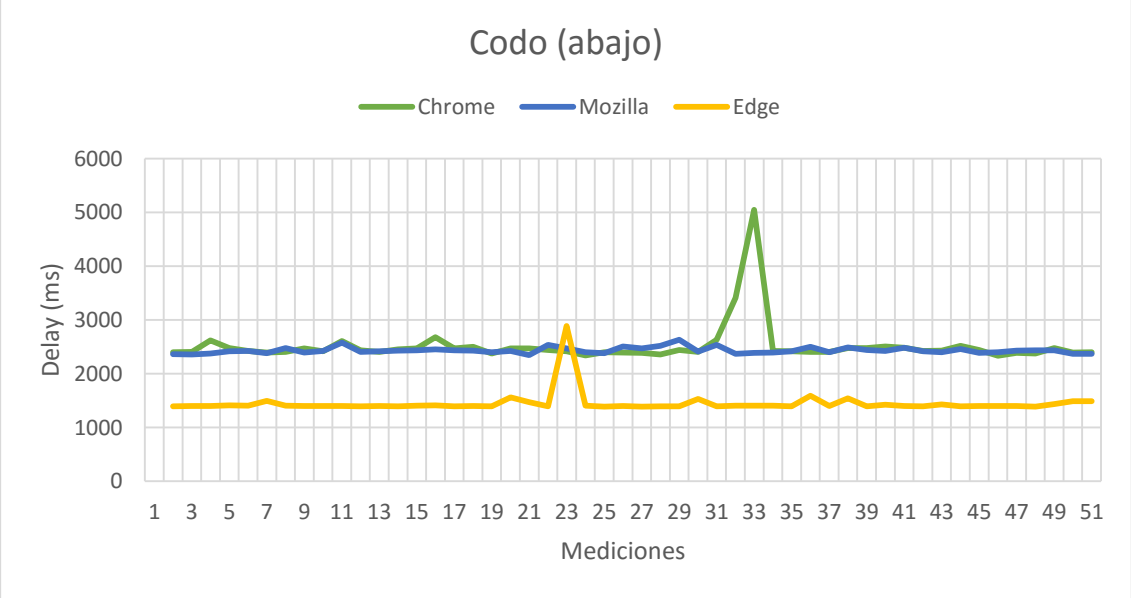


Figura 97: Movimiento del codo hacia abajo, red WAN, Ethernet v1. Elaboración propia.

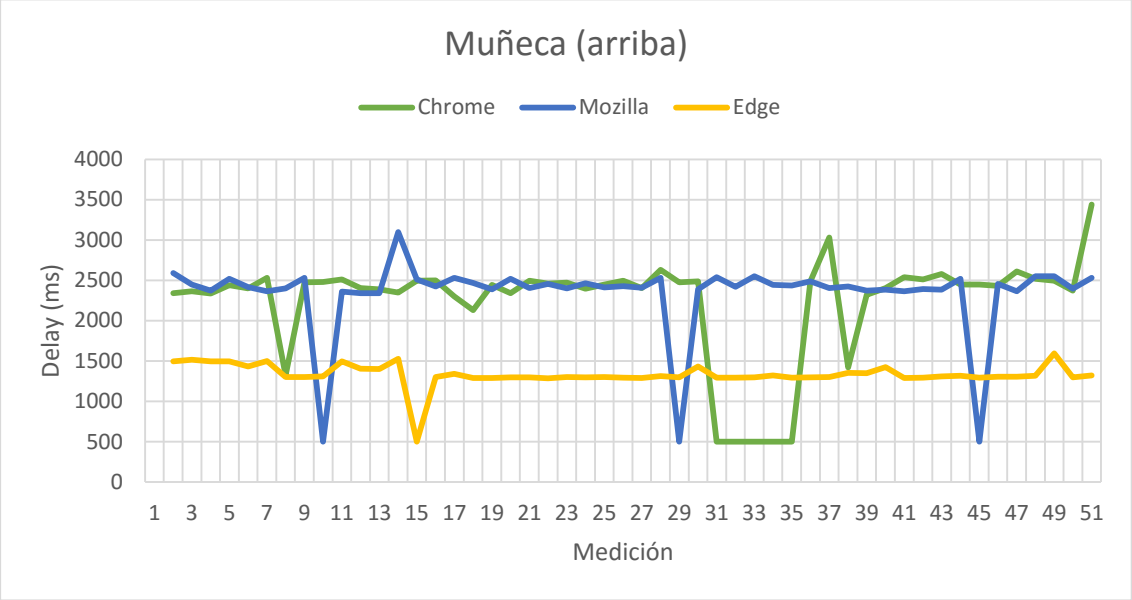


Figura 98: Movimiento de la muñeca hacia arriba, red WAN, Ethernet v1. Elaboración propia.

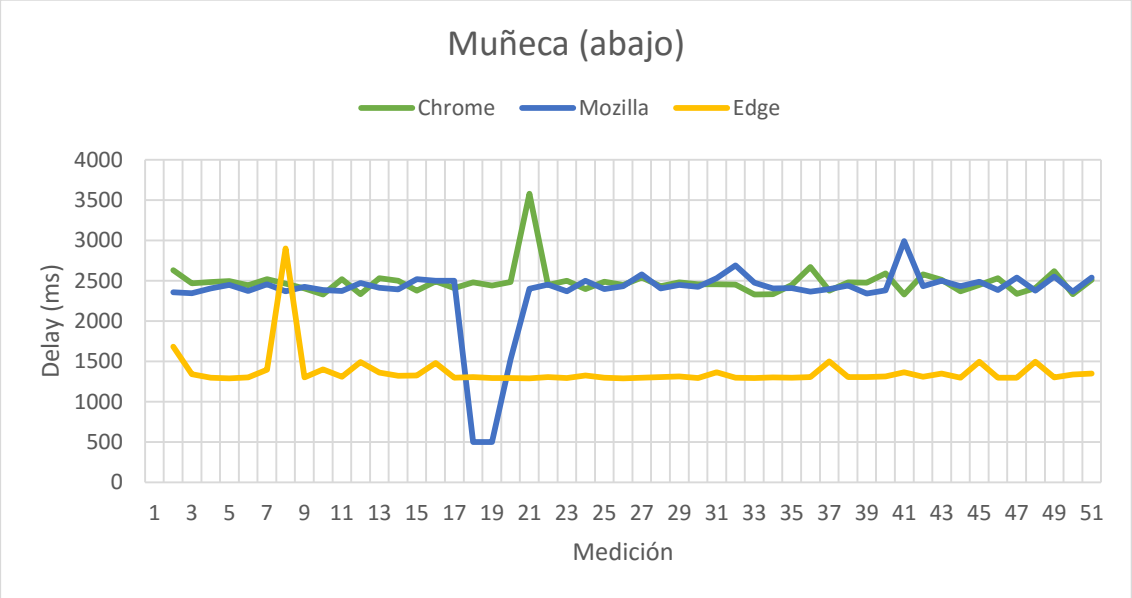


Figura 99: Movimiento de la muñeca hacia abajo, red WAN, Ethernet v1. Elaboración propia.

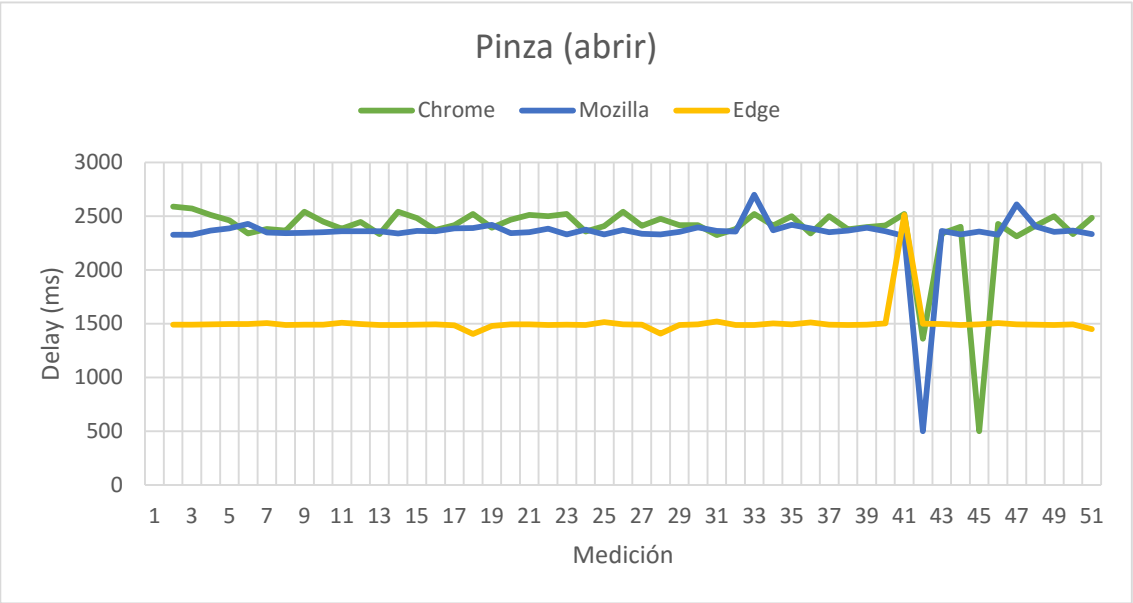


Figura 100: Movimiento al abrir la pinza, red WAN, Ethernet v1. Elaboración propia.

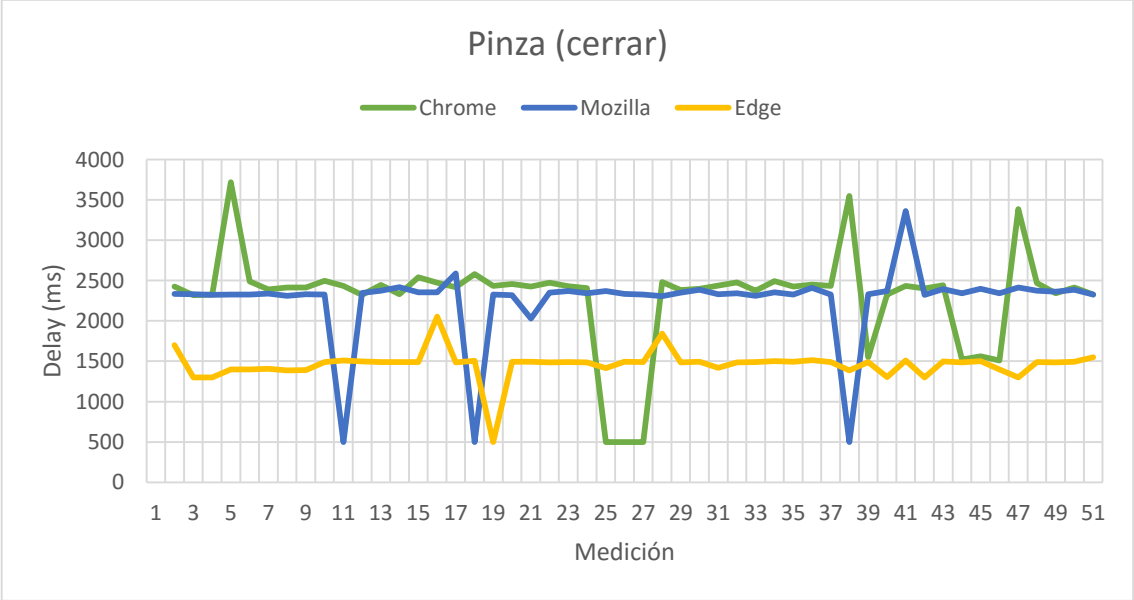


Figura 101: Movimiento al abrir la pinza, red WAN, Ethernet v1. Elaboración propia.

4. 2. 2. Ethernet v2.

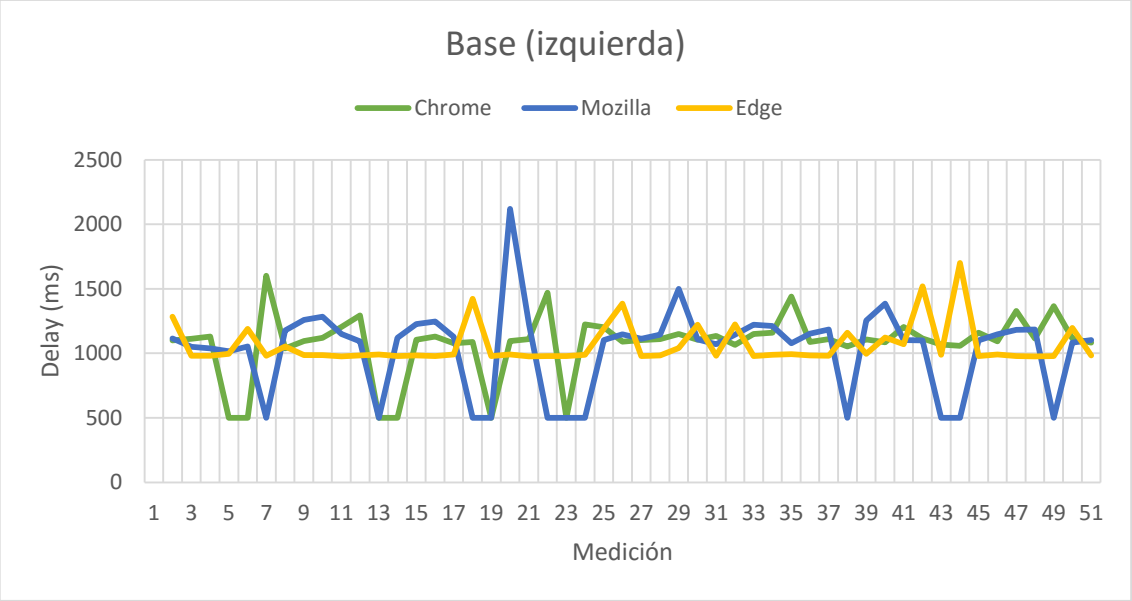


Figura 102: Movimiento de la base a la izquierda, red WAN, Ethernet v2. Elaboración propia.

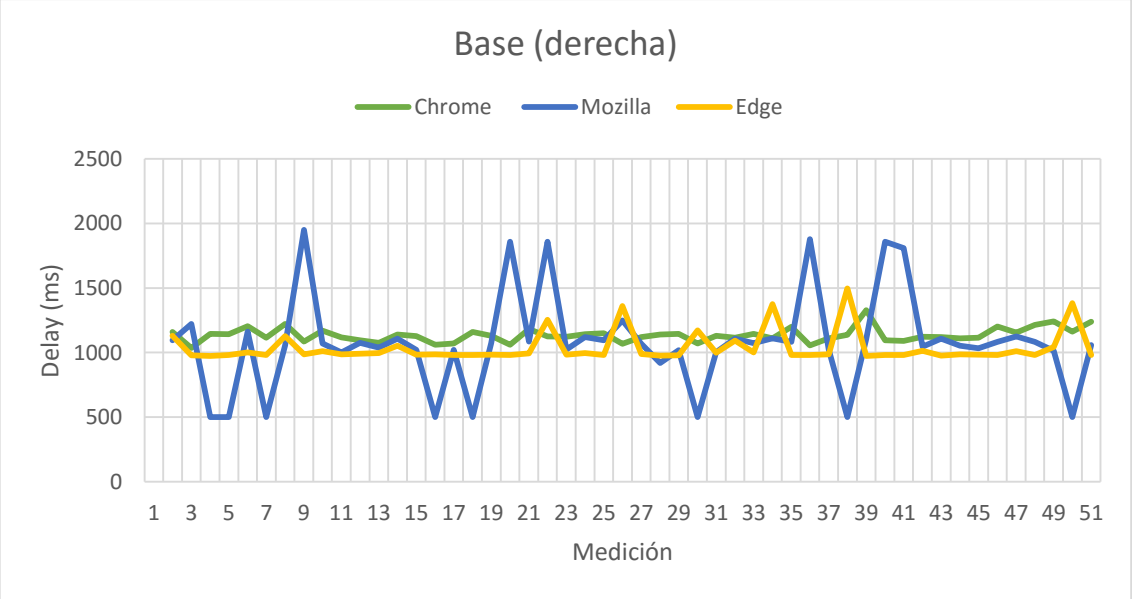


Figura 103: Movimiento de la base a la derecha, red WAN, Ethernet v2. Elaboración propia.

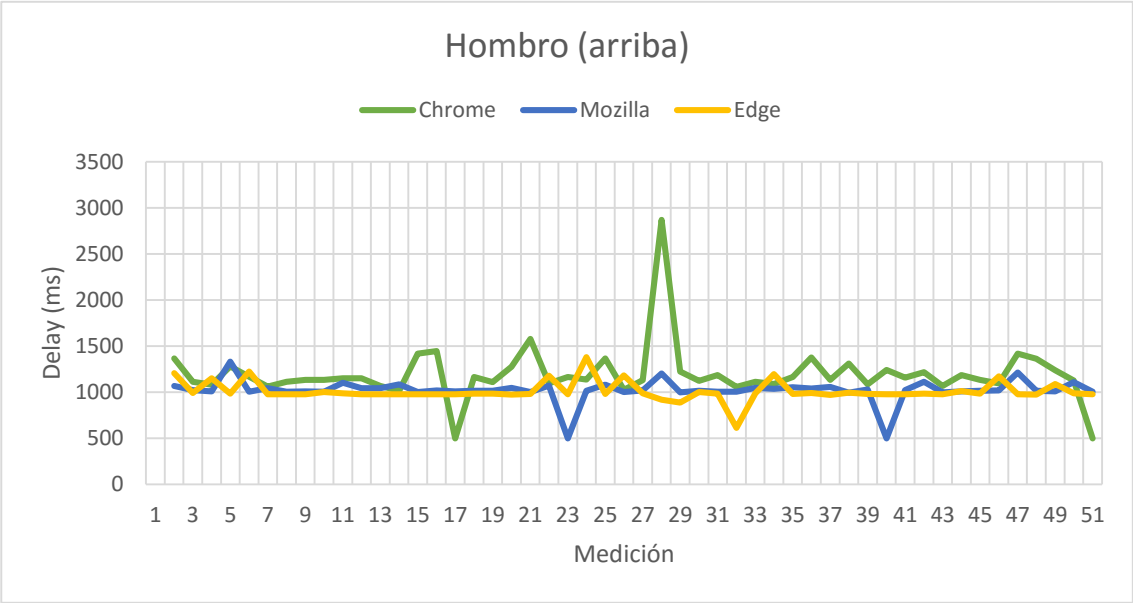


Figura 104: Movimiento del hombro hacia arriba, red WAN, Ethernet v2. Elaboración propia.

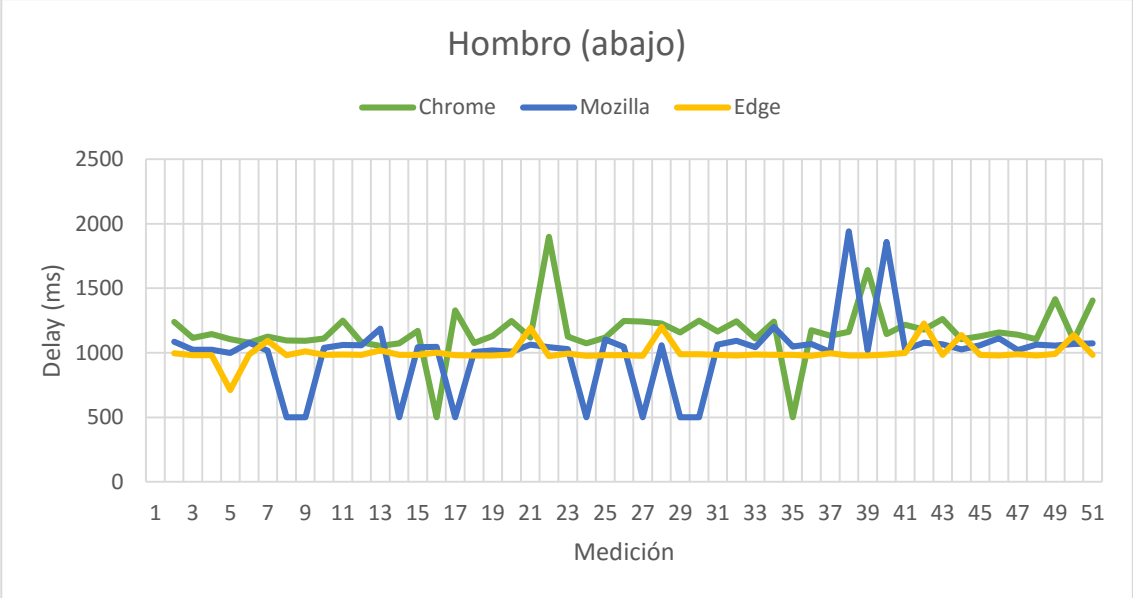


Figura 105: Movimiento del hombro hacia abajo, red WAN, Ethernet v2. Elaboración propia.

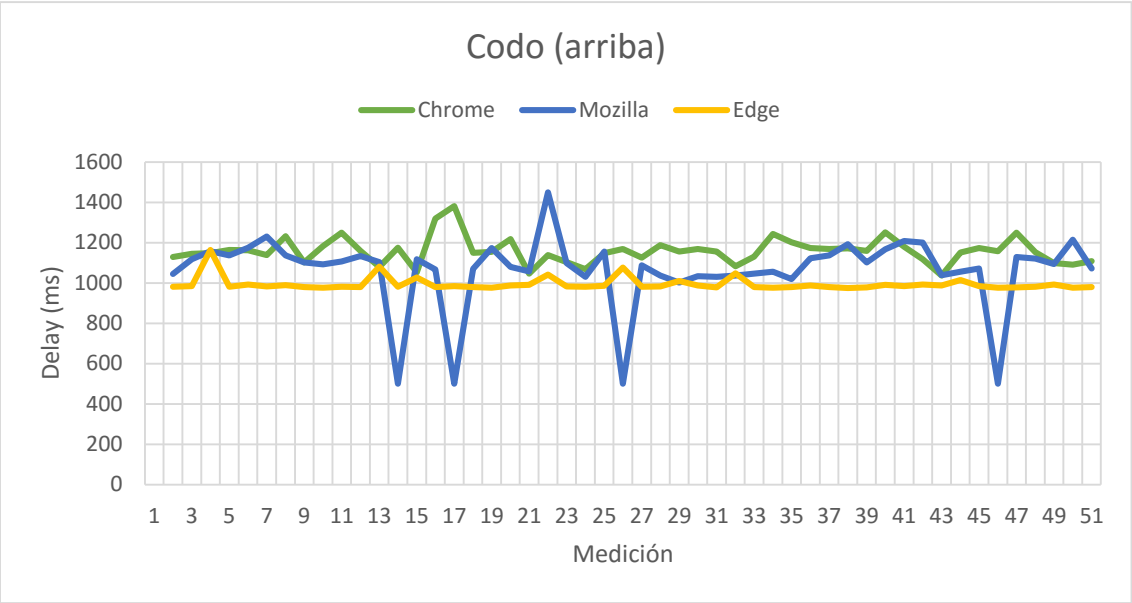


Figura 107: Movimiento del codo hacia arriba, red WAN, Ethernet v2. Elaboración propia.

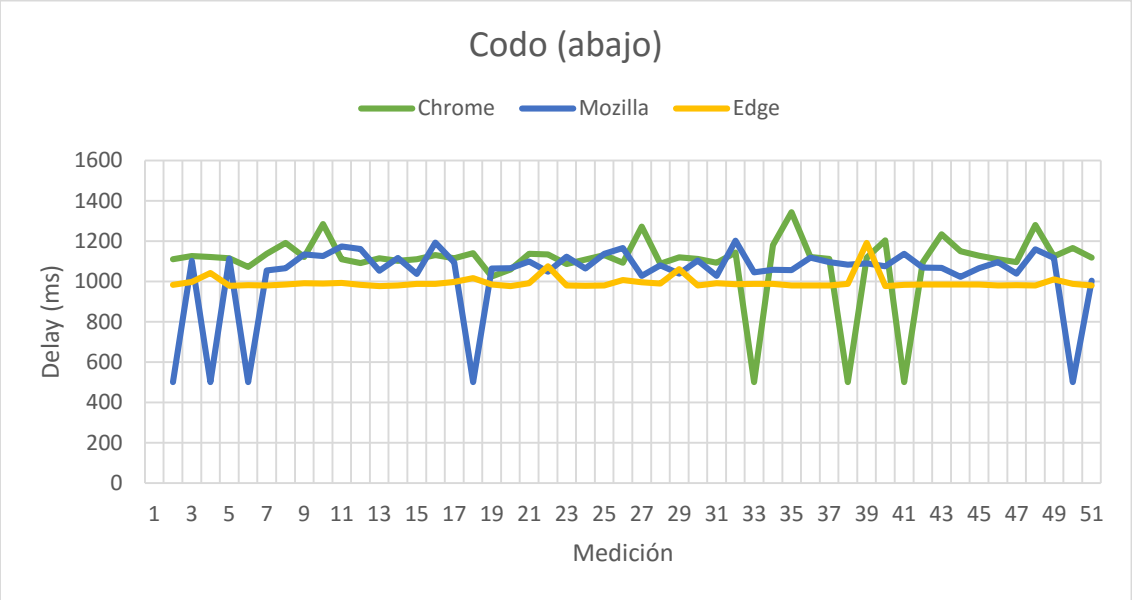


Figura 106: Movimiento del codo hacia abajo, red WAN, Ethernet v2. Elaboración propia.

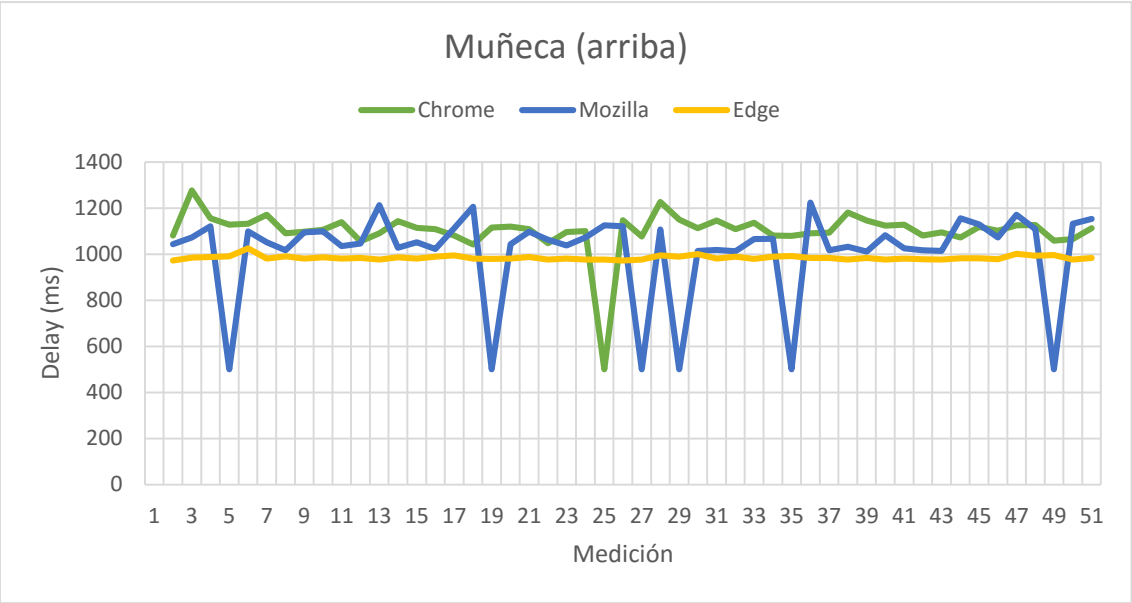


Figura 109: Movimiento de la muñeca hacia arriba, red WAN, Ethernet v2. Elaboración propia.

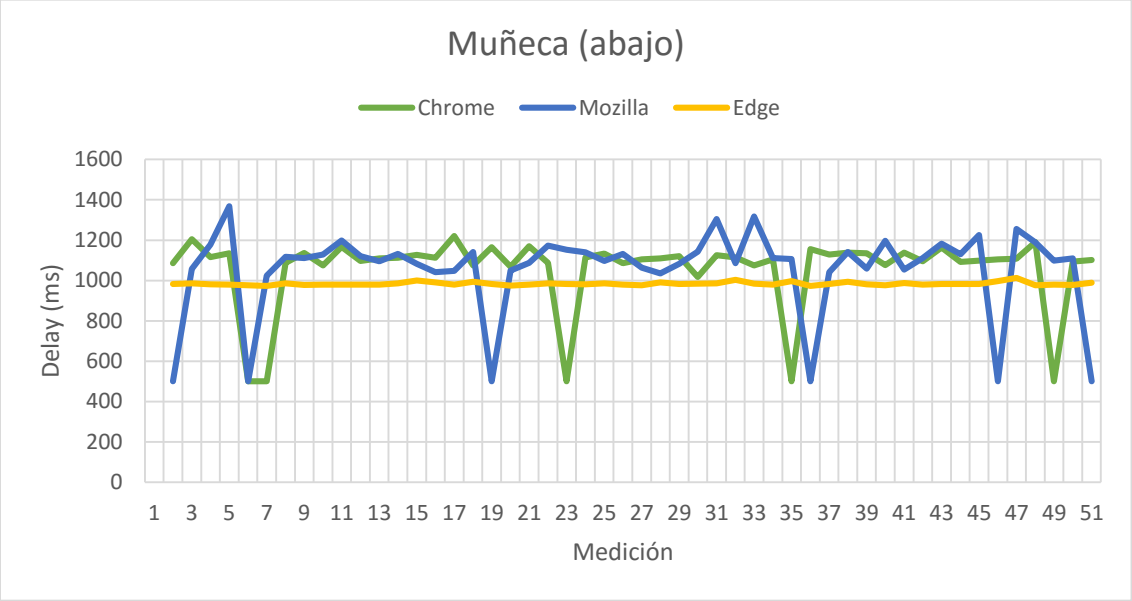


Figura 108: Movimiento de la muñeca hacia abajo, red WAN, Ethernet v2. Elaboración propia.

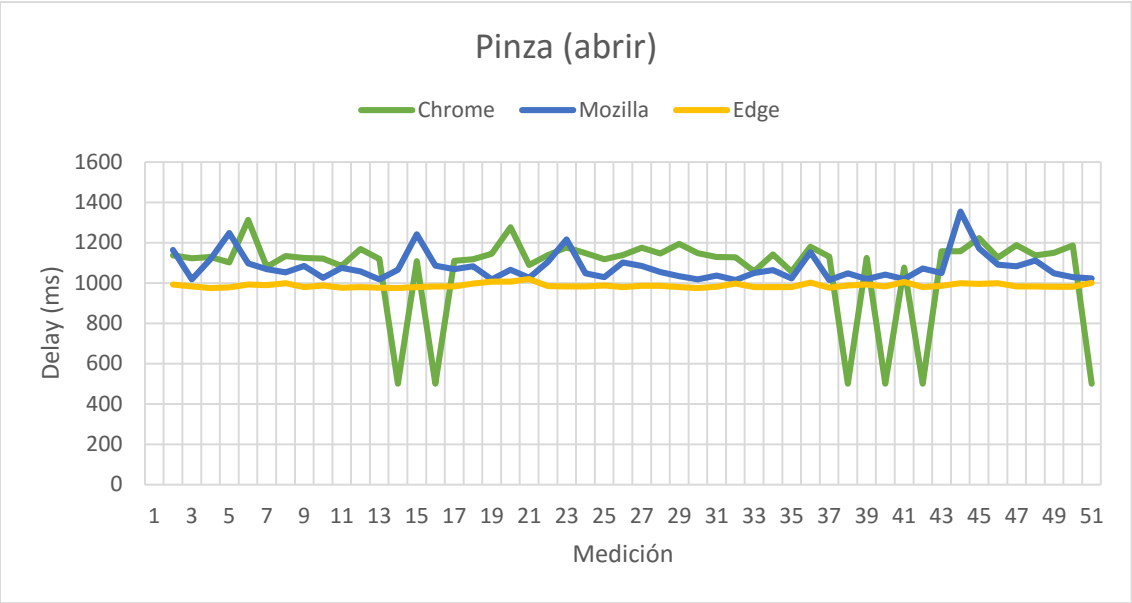


Figura 111: Movimiento al abrir la pinza, red WAN, Ethernet v2. Elaboración propia.

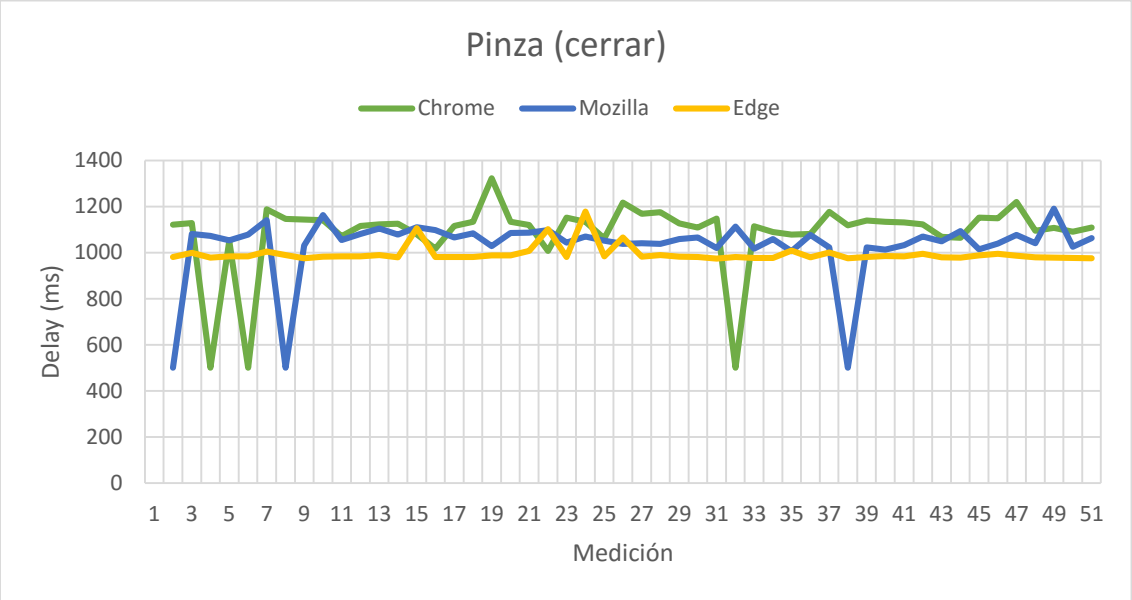


Figura 110: Movimiento al cerrar la pinza, red WAN, Ethernet v2. Elaboración propia.

4. 2. 3. Wi-Fi v1.

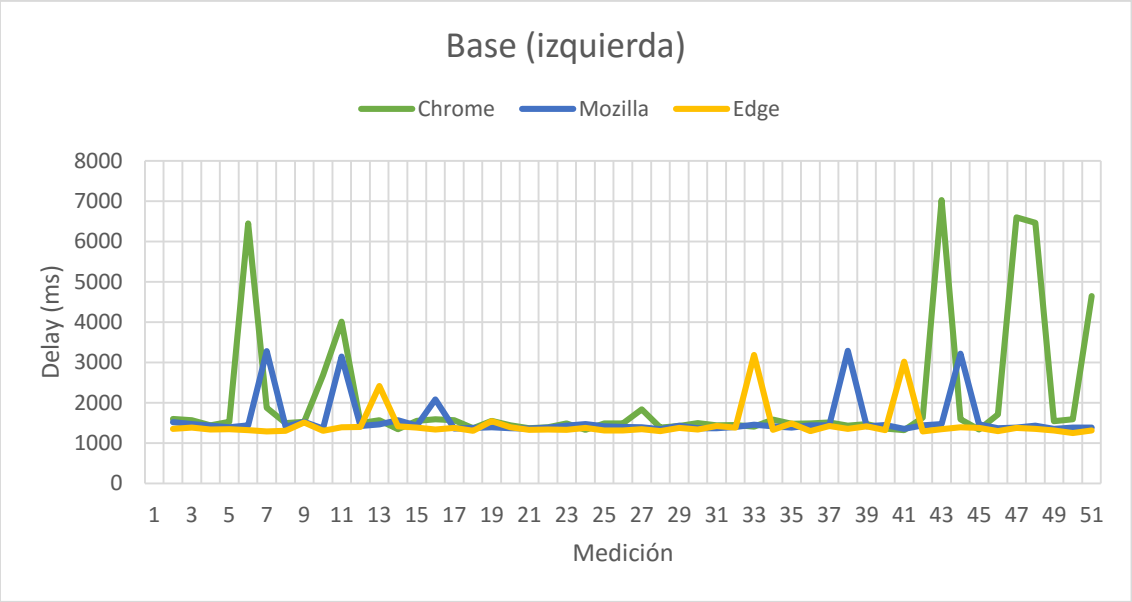


Figura 112: Movimiento de la base a la izquierda, red WAN, Wi-Fi v1. Elaboración propia.

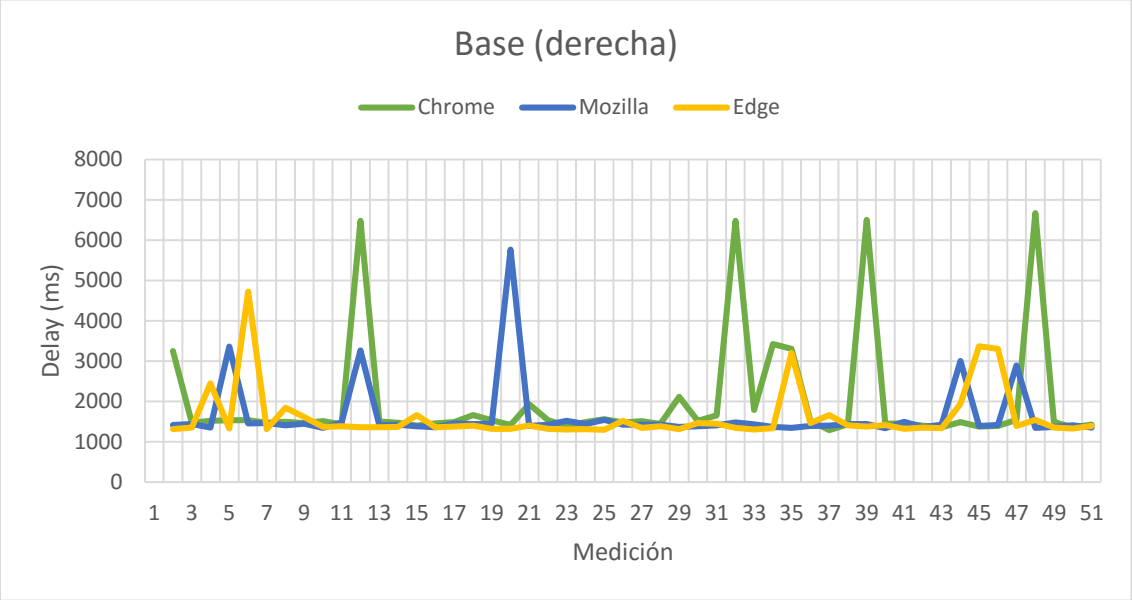


Figura 113: Movimiento de la base a la derecha, red WAN, Wi-Fi v1. Elaboración propia.

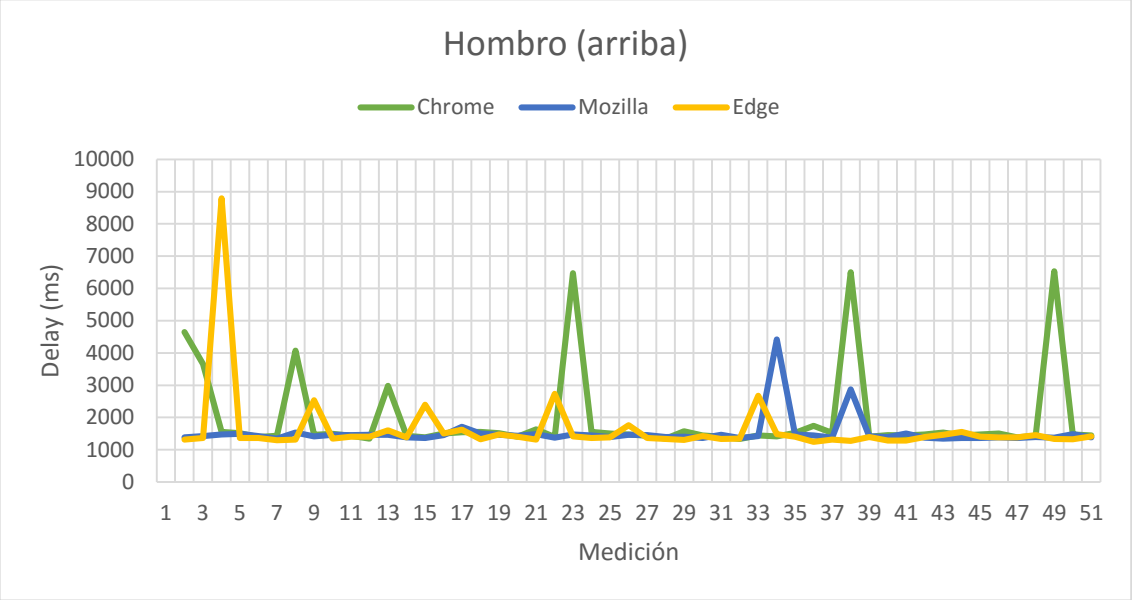


Figura 115: Movimiento del hombro hacia arriba, red WAN, Wi-Fi v1. Elaboración propia.

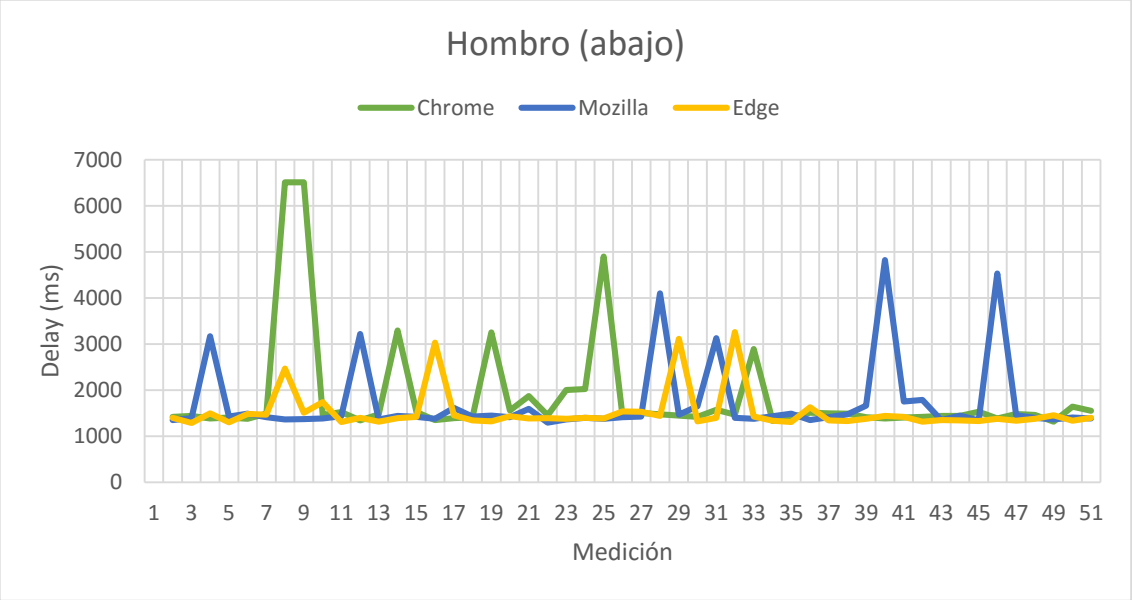


Figura 114: Movimiento del hombro hacia abajo, red WAN, Wi-Fi v1. Elaboración propia.

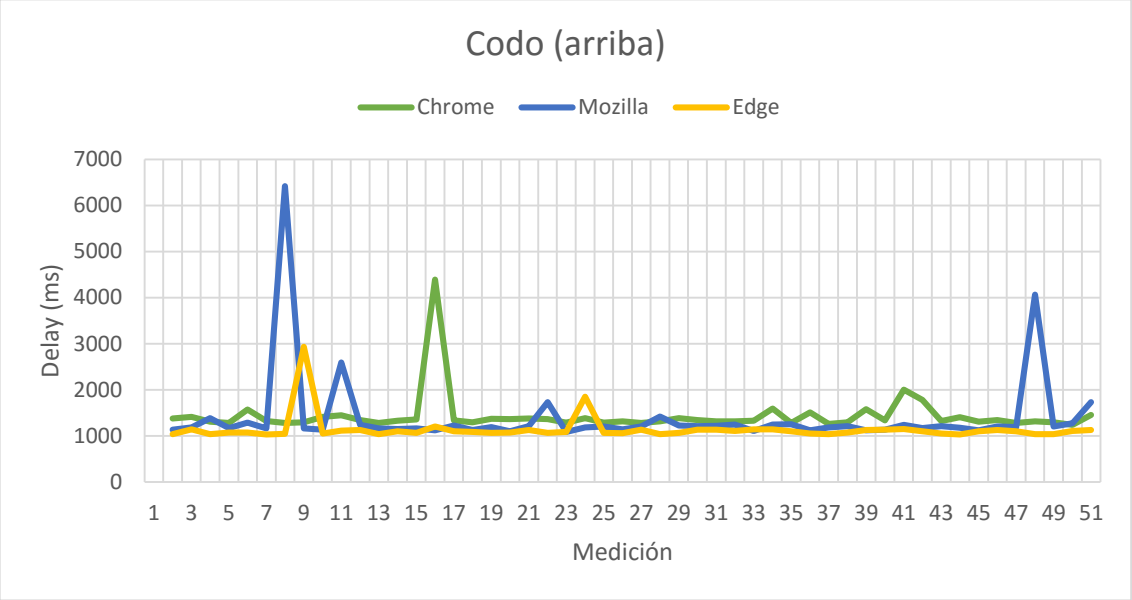


Figura 117: Movimiento del codo hacia arriba, red WAN, Wi-Fi v1. Elaboración propia.

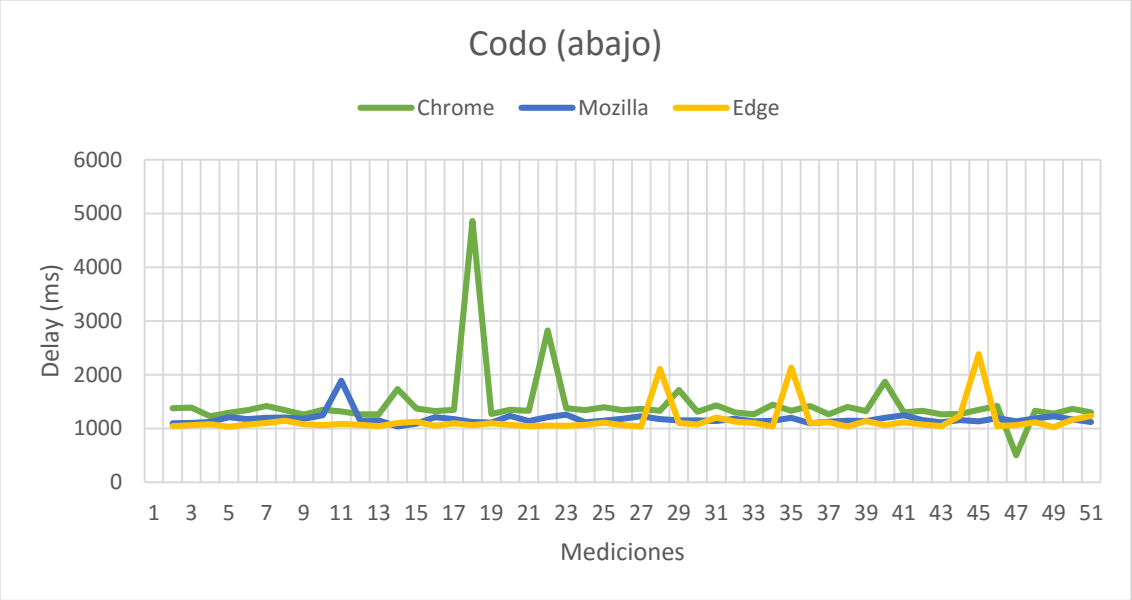


Figura 116: Movimiento del codo hacia abajo, red WAN, Wi-Fi v1. Elaboración propia.

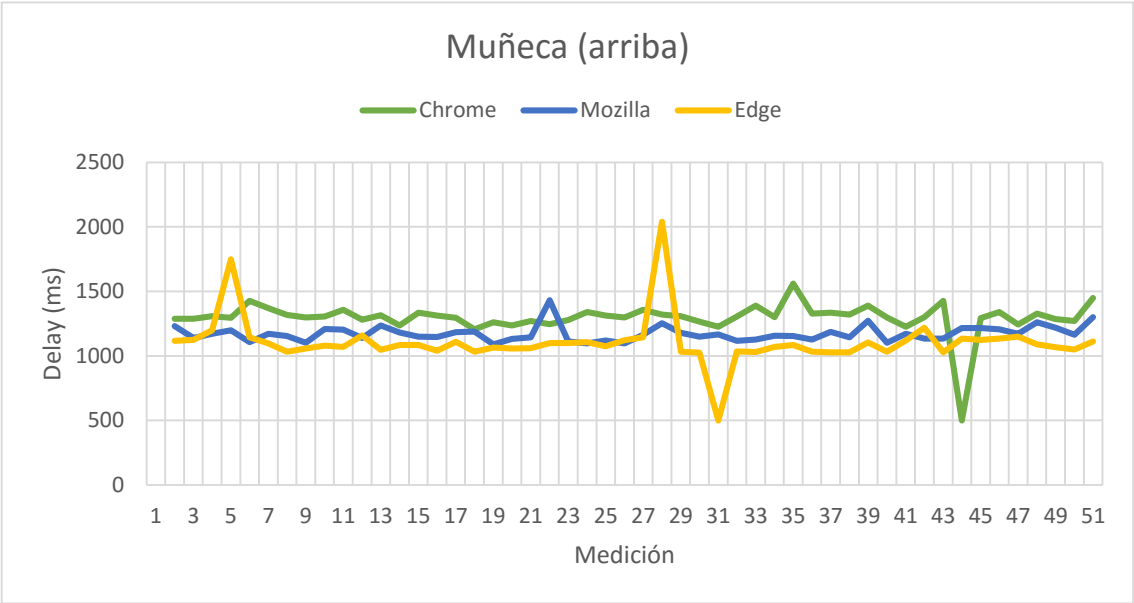


Figura 119: Movimiento de la muñeca hacia arriba, red WAN, Wi-Fi v1. Elaboración propia.

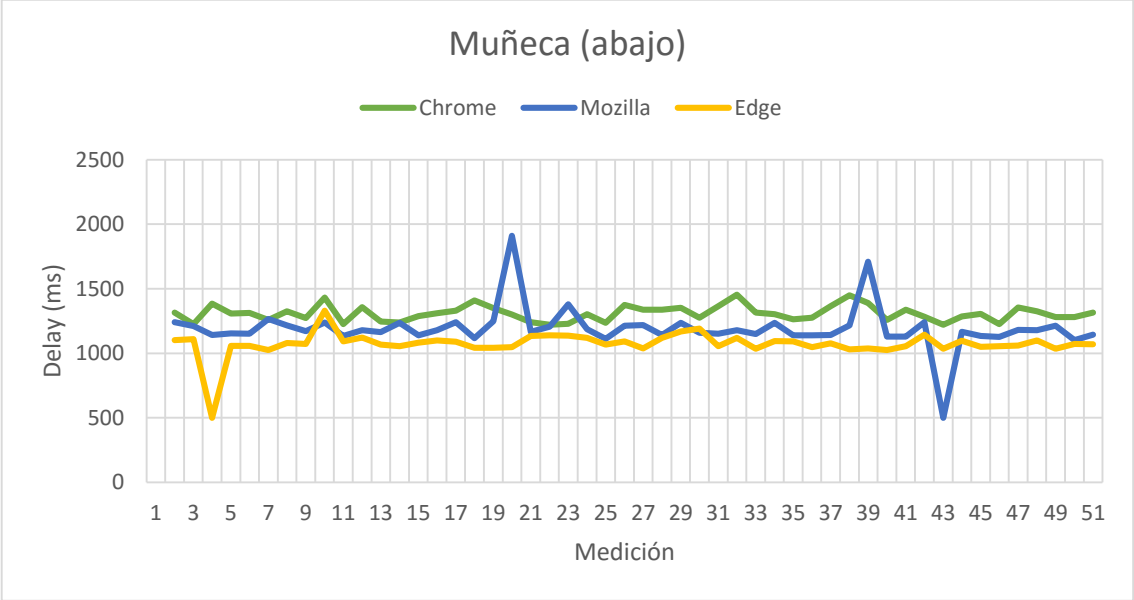


Figura 118: Movimiento de la muñeca hacia abajo, red WAN, Wi-Fi v1. Elaboración propia.

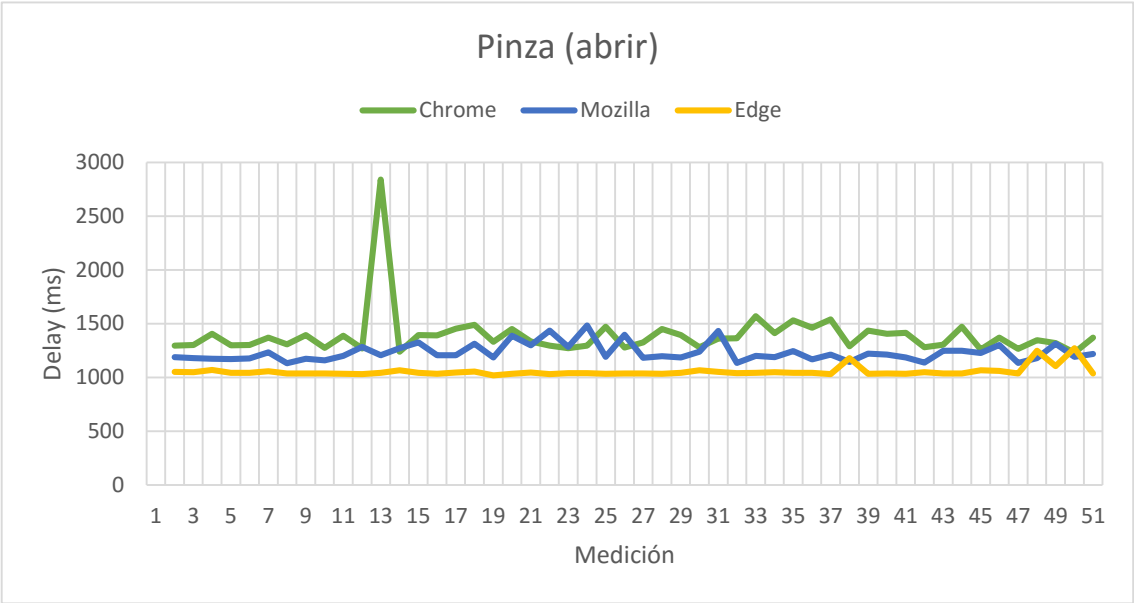


Figura 121: Movimiento al abrir la pinza, red WAN, Wi-Fi v1. Elaboración propia.

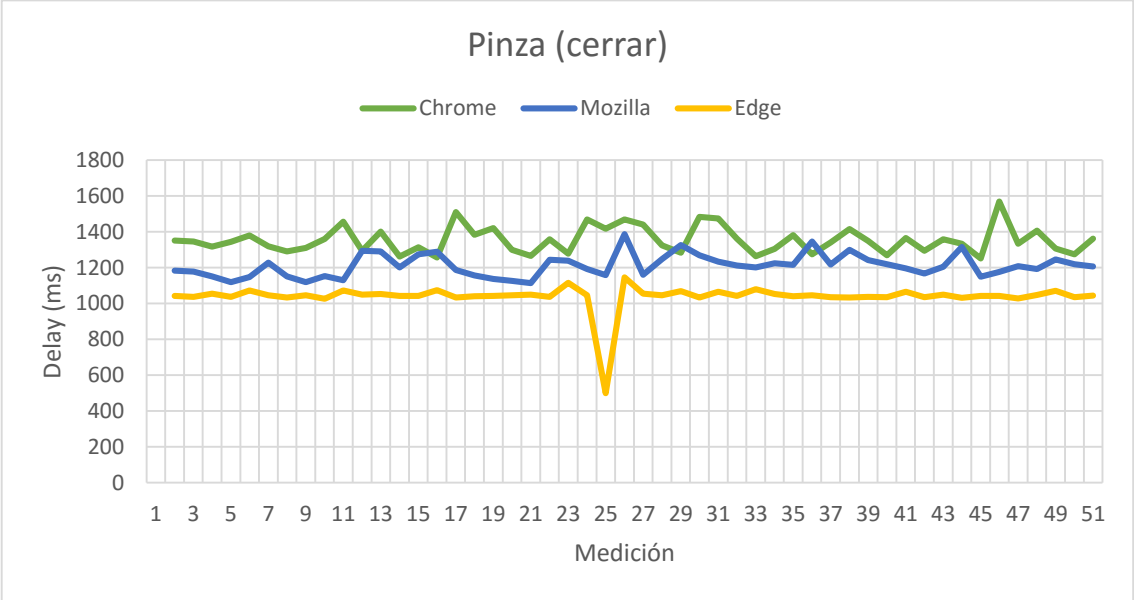


Figura 120: Movimiento al cerrar la pinza, red WAN, Wi-Fi v1. Elaboración propia.

4. 2. 4. Wi-Fi v2.

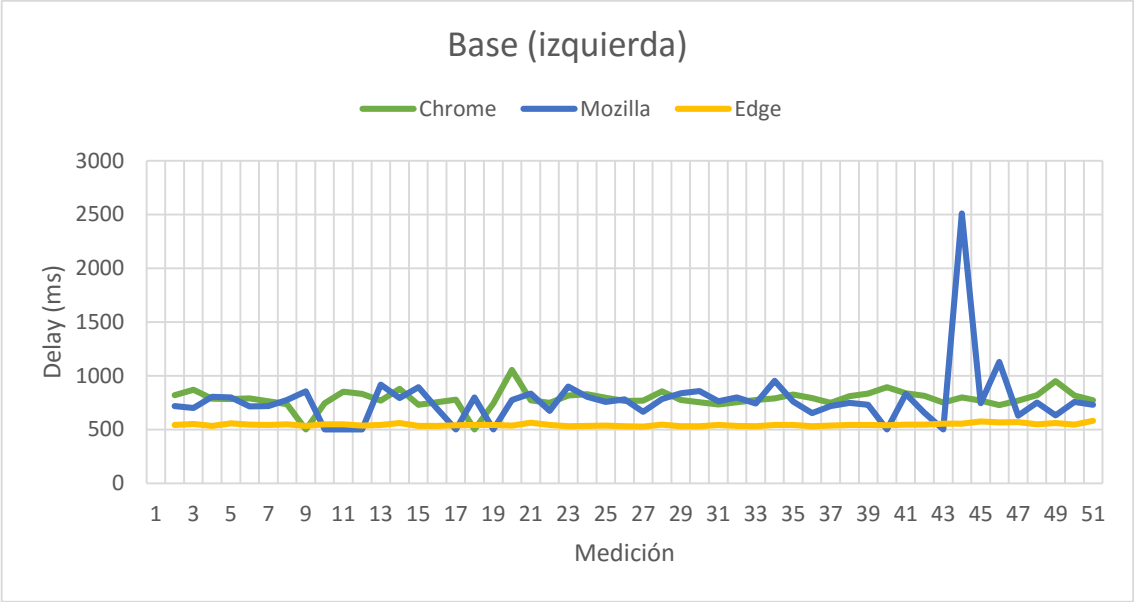


Figura 122: Movimiento de la base a la izquierda, red WAN, Wi-Fi v2. Elaboración propia.

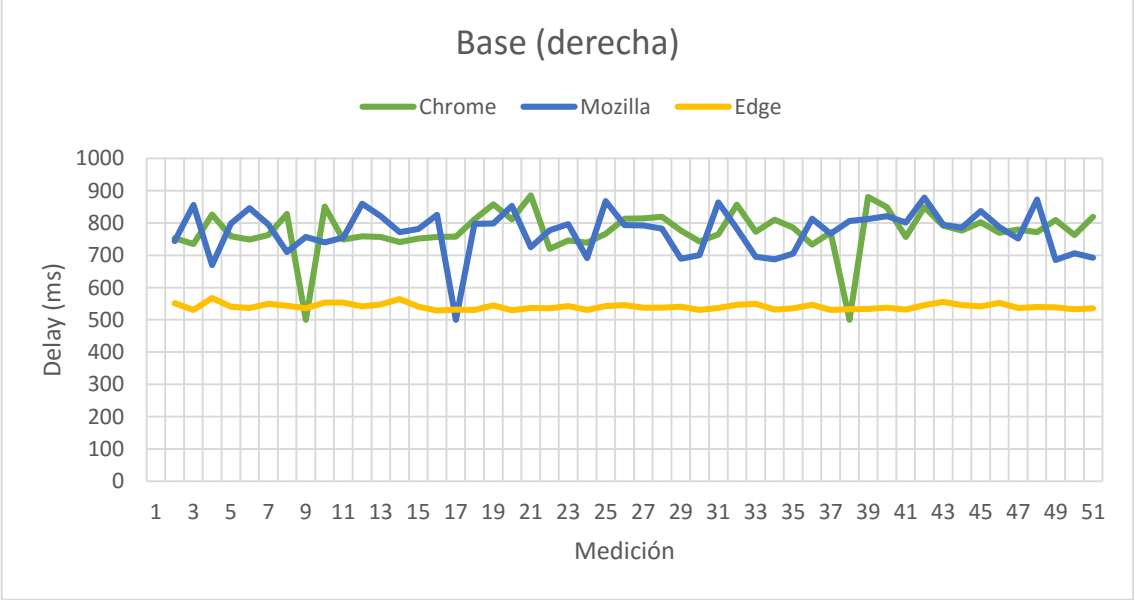


Figura 123: Movimiento de la base a la derecha, red WAN, Wi-Fi v2. Elaboración propia.

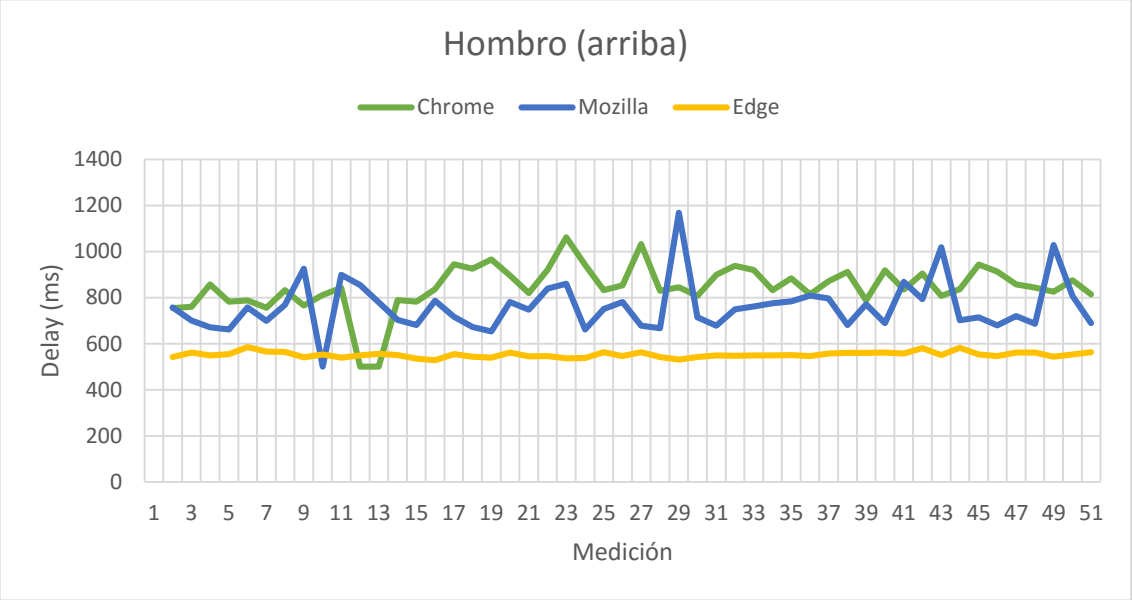


Figura 125: Movimiento del hombro hacia arriba, red WAN, Wi-Fi v2. Elaboración propia.

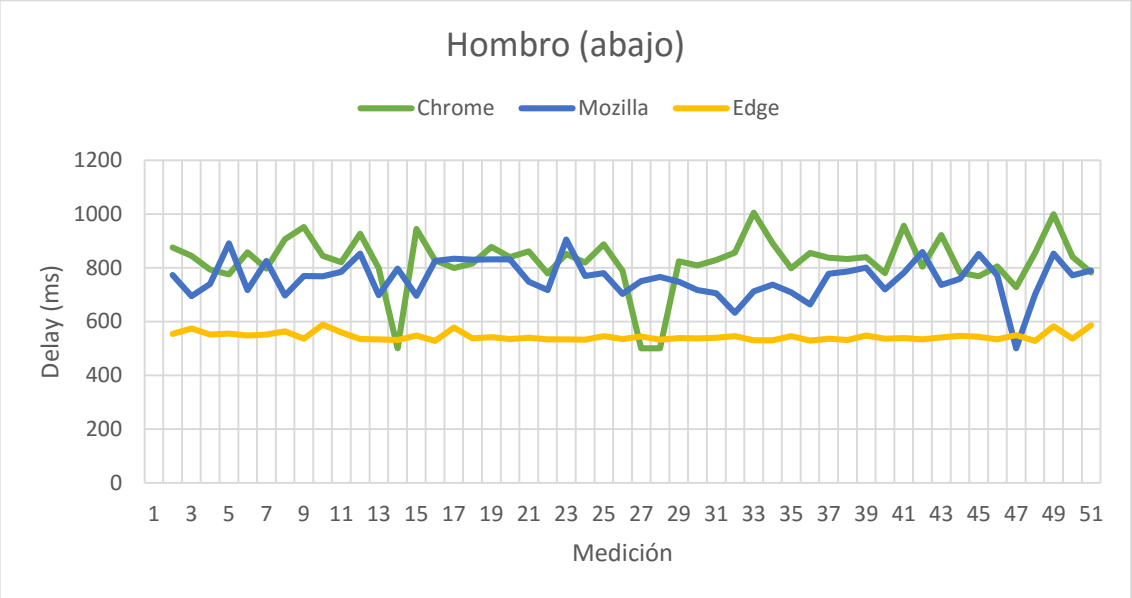


Figura 124: Movimiento del hombro hacia abajo, red WAN, Wi-Fi v2. Elaboración propia.

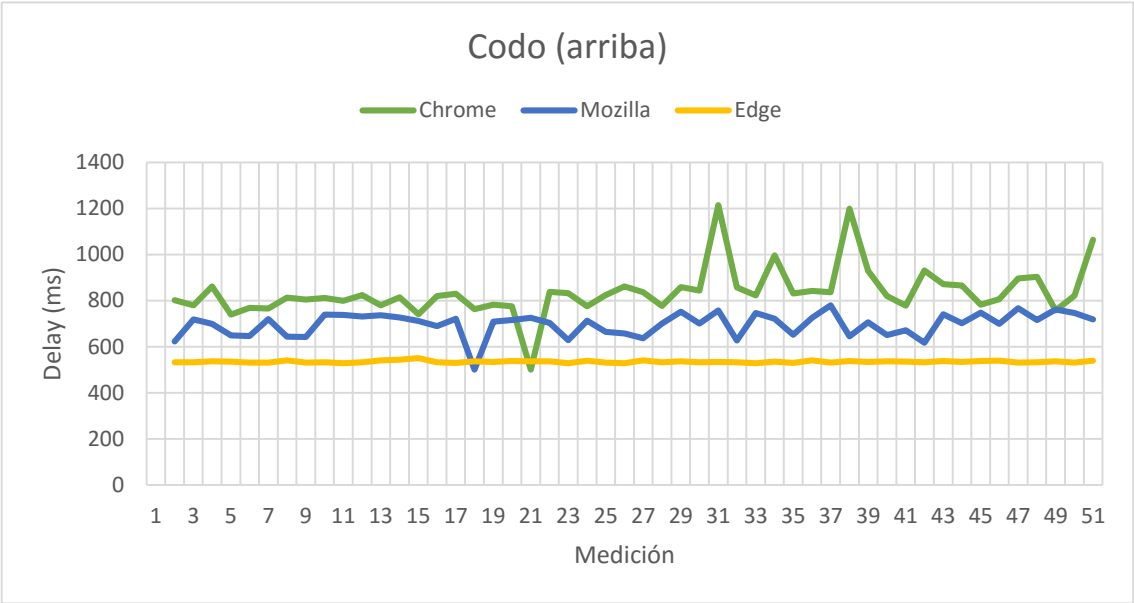


Figura 127: Movimiento del codo hacia arriba, red WAN, Wi-Fi v2. Elaboración propia.

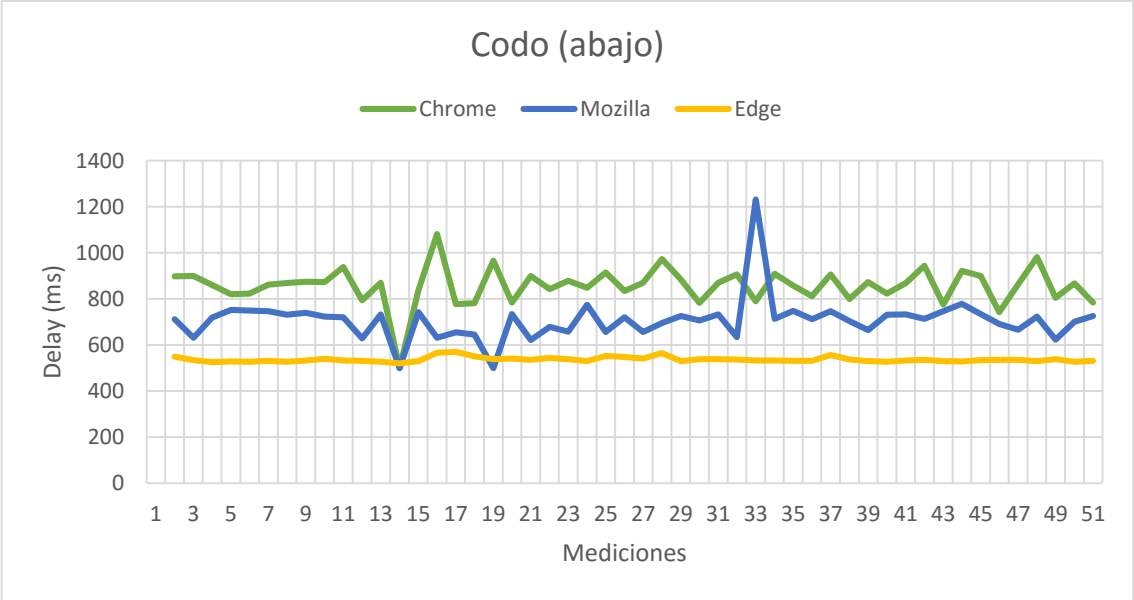


Figura 126: Movimiento del codo hacia abajo, red WAN, Wi-Fi v2. Elaboración propia.

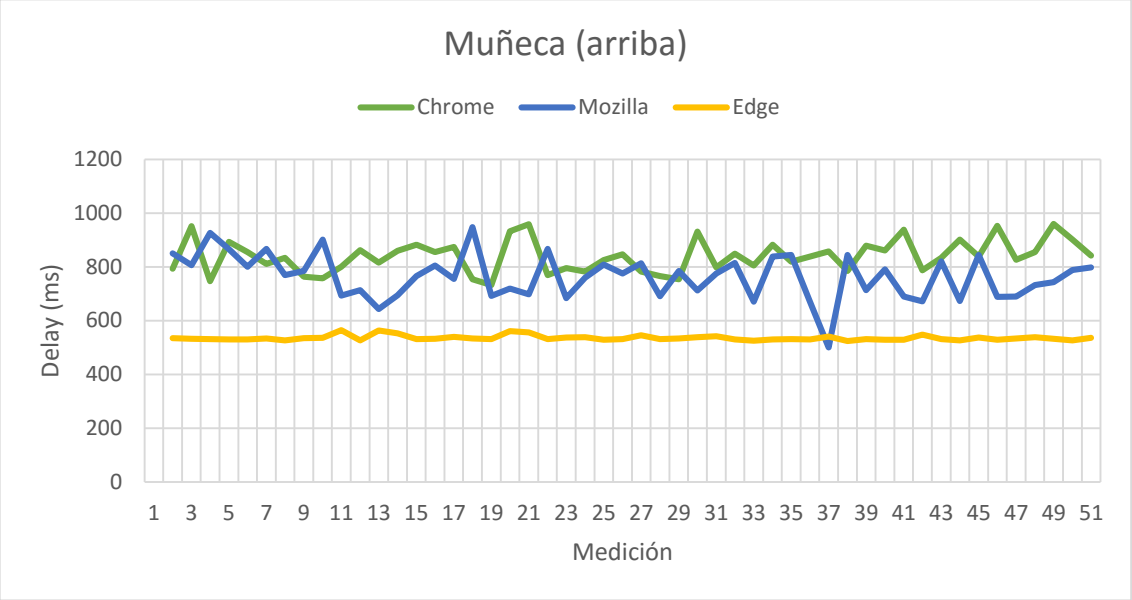


Figura 128: Movimiento de la muñeca hacia arriba, red WAN, Wi-Fi v2. Elaboración propia.

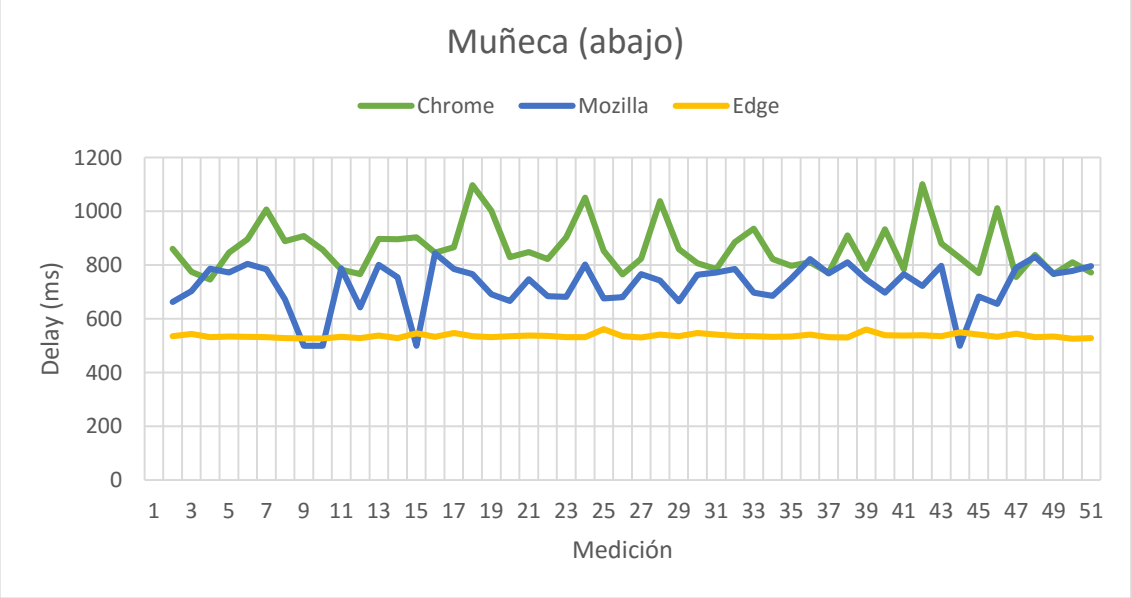


Figura 129: Movimiento de la muñeca hacia arriba, red WAN, Wi-Fi v2. Elaboración propia.

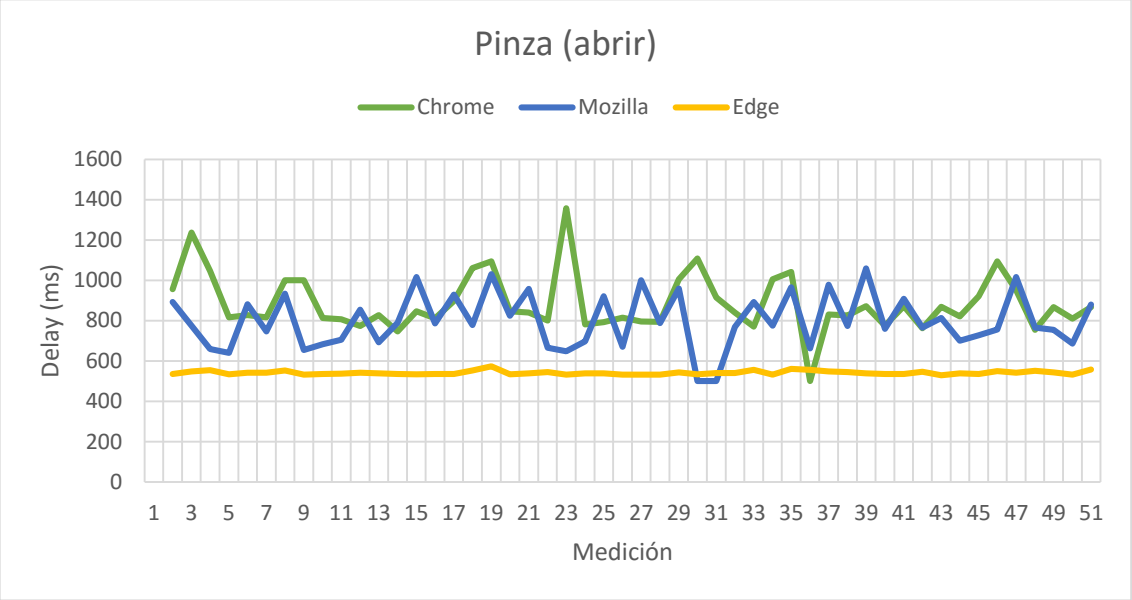


Figura 130: Movimiento al abrir la pinza, red WAN, Wi-Fi v2. Elaboración propia.

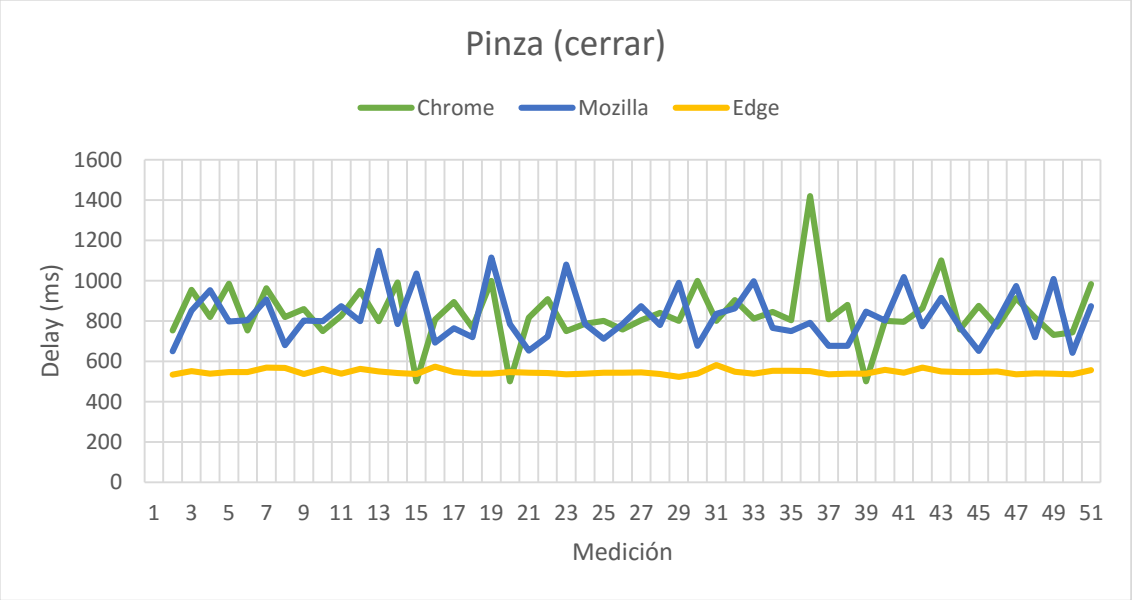


Figura 131: Movimiento al cerrar la pinza, red WAN, Wi-Fi v2. Elaboración propia.

V. Conclusiones.

A partir del trabajo realizado se encontró que el Shield W5100 presenta inconvenientes de compatibilidad con las versiones más nuevas de HTML lo cual obliga al programador a sacrificar diseño por una mayor velocidad, este sacrificio en diseño puede ocasionar que las páginas creadas en dicho módulo sean más simples y poco intuitivas llegando a ser incomodas para el usuario final. Además de eso es incompatible con IPv6 así que muy pronto será obsoleto. Por otro lado, si lo que se desea es crear páginas para aplicaciones que no requieran demasiada velocidad o un diseño muy avanzado el W5100 será una opción adecuada gracias a su bajo precio y la facilidad de su programación.

El NodeMCU resultó ser más veloz y debido a sus pequeñas dimensiones y al no requerir de conexión por cable se vuelve el indicado para aplicaciones en las que se necesita optimizar espacios. Tiene problemas de compatibilidad con IPv6, pero si es compatible con versiones más modernas de HTML por lo que permite un diseño de páginas web más complejas e intuitivas, su costo es accesible y sin duda es una mejor alternativa que el W5100, aunque no se recomienda su uso por personas que no posean ciertos conocimientos de programación y del idioma inglés pues la documentación en español es escaza.

Ninguno de los dispositivos parece presentar problemas de sobrecalentamiento y trabajaron perfectamente por periodos superiores a las 24 horas seguidas. Es destacable que el NodeMCU no presento variaciones importantes en su velocidad al usarse en redes WAN, mientras que el W5100 sí tuvo problemas como perdida de conexión, elevados tiempos de carga o la no detección de los eventos onpointerup y onClick ocasionando que el robot no detuviera su movimiento aun cuando se pulsaba el botón de paro de emergencia, por lo que se recomienda ampliamente no usar el Ethernet Shield en redes WAN.

Los resultados también arrojaron que es importante seleccionar adecuadamente el buscador utilizado para abrir la página web. En ese sentido, Microsoft Edge se presenta como la mejor elección al ser el más rápido y prácticamente no presentar problemas de conexión o de estabilidad, Google Chrome es una opción menos estable pero igualmente rápida y Mozilla Firefox resulto ser la peor alternativa al presentar una baja velocidad de carga y muchos problemas de estabilidad y conexión.

Las pruebas realizadas en un dispositivo móvil arrojaron que no es recomendable utilizarlo para acceder a la página pues no se detectaran adecuadamente los eventos onpointer, es más recomendable la creación de una aplicación móvil que permita un control adecuado de estos eventos.

En general, el sistema propuesto se puede utilizar de manera didáctica para enseñar conceptos sobre redes, telecomunicaciones y telecontrol, siempre siguiendo la recomendación de utilizar un servicio de Internet que permita conexión interna y externa con el servidor.

Bibliografía.

Barbancho, J., Benjumea, J., Rivera, O., Ropero, J., Sánchez, G., Sivianes, F., & Romero, M. (2014). Medios de transmisión. En *Redes locales (segunda ed.)* (págs. 70-80). Llanera, Asturias: Paraninfo.

Barbancho, J., Benjumea, J., Rivera, O., Ropero, J., Sánchez, G., Sivianes, F., & Romero, M. d. (2014). *Redes locales (segunda ed.).* Llanera, Asturias: Paraninfo.

Barrenechea, I. (2011). *Diseño de una red LAN inalambrica para una empresa de Lima (tesis de pregrado).* Lima, Perú: Pontificia Universidad Católica del Perú .

Betancur, L. (2011). Redes de área corporal. Una perspectiva al futuro desde la investigación. . *Revista Sistemas y Telemática, 9*(16), 11-30.

Cisneros Limón, R. (2006). *Modelo matemático de un robot paralelo de seis grados de libertad (tesis de pregrado).* Cholula, Puebla: Universidad de las Américas Puebla.

Gómez, J. (2008). Normativa IEEE 802.11. En *Guía de campo de WiFi* (págs. 20-21). México: Alfaomega Ra-Ma.

Gómez, J. (2008). Redes inalámbricas de área local . En *Guía de campo de WiFi* (págs. 8-9). México: Alfaomega Ra-Ma.

Huidobro, J. (2004). El método CSMA/CD. En *Manual de Telecomunicaciones* (págs. 166-170). México: Alfaomega Ra-Ma.

Huidobro, J., & Roldán, D. (2006). Pila de protocolos. En *Comunicaciones en redes WLAN: WiFi, VoIP, multimedia y seguridad.* (pág. 179). México: Limusa.

Huidobro, J., & Roldán, D. (2006). Principios básicos de 802.11. En *Comunicaciones en redes WLAN: Wifi, VoIP, multimedia y seguridad.* (págs. 169-171). México: Limusa.

Martin, J., Kavanagh, K., & Leben, J. (1994). CSMA/CD Architectural Model. En *Local area networks: architectures and implementations (2nd ed)* (págs. 173-174). Englewood Cliffs, New Jersey: P T R Prentice Hall.

Martin, J., Kavanagh, K., & Leben, J. (1994). Local Area Networks. En *Local area networks: architectures and implementations (2nd ed)* (págs. 3-5). Englewood Cliffs, New Jersey: P T R Prentice Hall.

Rábago, F. (2008). Sistemas de cableado y tipos de redes. En *Redes locales. Edición 2008* (págs. 167-173). Madrid, España: Anaya multimedia .

Ramos, A., Barbero, C., Fernández, Y., Daswani, D., & Marugán, D. (2015). Protocolos y estándares. En *Hacking práctico de redes wifi y rediofrecuencia* (págs. 21-24). Bogotá, Colombia: Ra-Ma.

Steren. (s.f.). *Manual de instrucciones VO608.* México: Electrónica Steren.

Velte, T. J., & Velte, A. T. (2008). *Manual de Cisco. Cuarta Edición. (Carlos Cordero & Jorge Pineda, trad.).* México, D.F: McGraw-Hill Interamericana.

Vigil Sanabria, G. (2009). *Configuración matemática de un brazo robot con uniones de Cardan de trayectoria específica (tesis de maestría).* Ciudad Universitaria, México D.F.: Universidad Nacional Autónoma de México.

Referencias electrónicas.

Alejandro, C., & Miguel, L. (10 de Mayo de 2017). *Grados de libertad*. Recuperado el 6 de Abril de 2018, de Inteligencia artificial: https://inteligenciartificiallmca.wordpress.com/2017/05/10/2-5-grados-de-libertad-de-un-robot/

Arduino. (s.f.). *Arduino Ethernet Shield without PoE module*. (Arduino) Recuperado el 20 de Julio de 2018, de Arduino: https://store.arduino.cc/usa/arduino-ethernet-shield-without-poe-module

Ashton, K. (22 de Junio de 2009). *That 'Internet of Things' Thing*. (RFID Journal) Recuperado el 4 de Abril de 2018, de RFID Journal: https://www.rfidjournal.com/articles/view?4986

Cisco. (s.f.). *Ethernet*. (Cisco) Recuperado el 2 de Junio de 2018, de Cisco: https://www.cisco.com/c/es_mx/tech/lan-switching/ethernet/index.html

Crespo, E. (23 de Marzo de 2015). *Shields para Arduino*. (Aprendiendo Arduino) Recuperado el 15 de Julio de 2018, de Aprendiendo Arduino: https://aprendiendoarduino.wordpress.com/2015/03/23/shields-para-arduino/

El Universal. (5 de Agosto de 2013). El Universal. *Sonda Curiosity cumple un año en Marte*. Cartagena, Colombia: El Universal. Obtenido de http://www.eluniversal.com.co/ciencia/sonda-curiosity-cumple-un-ano-en-marte-130254-MTEU218004

Evans, D. (Abril de 2011). *Cisco Internet Business Solutions Group (IBSG).* Recuperado el 4 de Abril de 2018, de Cisco: https://www.cisco.com/c/dam/global/es_mx/solutions/executive/assets/pdf/internet-of-things-iot-ibsg.pdf

Glosario de Riego. (2018). *Telecontrol*. Recuperado el 2 de Mayo de 2018, de Glosario de Riego: https://www.riego.org/glosario/telecontrol/

González, V. (2002). *Estructura de un robot industrial*. Recuperado el 6 de Abril de 2018, de Fundamentos de robótica: http://platea.pntic.mec.es/vgonzale/cyr_0204/cyr_01/robotica/sistema/morfologia.htm

Historia de la informatica. (2 de Diciembre de 2010). *Historia de las Redes Inalámbricas*. (Universidad Politécnica de Valéncia) Recuperado el 2 de Junio de 2018, de Blog Historia de la Informatica: http://histinf.blogs.upv.es/2010/12/02/historia-de-las-redes-inalambricas/

IEEE. (2018). *IEEE Standards Association*. (IEEE) Recuperado el 6 de Junio de 2018, de IEEE: https://standards.ieee.org/content/ieee-standards/en/standard/

INFOOTEC. (s.f.). *Arduino UNO R3*. (INFOOTEC.NET) Recuperado el 20 de Julio de 2018, de INFOOTEC.NET: https://www.infootec.net/arduino/

Jaime. (17 de Abril de 2016). *Master class sobre robots industriales*. Recuperado el 4 de Abril de 2018, de Robiteck Sistemas: https://robiteck.com/master-class-robots-industriales/

Laborda, J. (1 de Octubre de 2016). *Introducción al ESP8266 y NodeMCU*. (GitHub) Recuperado el 21 de Julio de 2018, de GitHub: https://github.com/jaimelaborda/Planta-Twittera/wiki/1.-Introducci%C3%B3n-al-ESP8266-y-NodeMCU

Márquez Aguilera, M., & Giron Bobadilla, H. (2018). *Grados de libertad*. Recuperado el 6 de Abril de 2018, de Inteligencia artificial: https://freedoomforlife.wordpress.com/grados-de-libertad/

Martínez, E. (20 de Abril de 2012). *La historia del lenguaje de Internet: TCP/IP*. (Eveliux.com) Recuperado el 2 de Mayo de 2018, de Eveliux: http://www.eveliux.com/mx/La-historia-del-lenguaje-de-Internet-TCP/IP.html

Morente, L. (15 de Abril de 2017). Expansión. *Todo lo que debes saber sobre Da Vinci, el robot quirúrgico*. Unidad Editorial Información General, S.L.U. Obtenido de http://www.expansion.com/tecnologia/2017/04/15/58f24ada22601d67308b460b.html

Noticias Mutimedia. (3 de Junio de 2015). *Telepresencia, ¿para qué?* (Multimedia) Recuperado el 6 de Abril de 2018, de Multimedia: http://www.multimedia.com.mx/blog/index.php/colaboracion/telepresencia/50-telepresencia-para-que

Noticias Mutimedia. (28 de Abril de 2017). *¿Sabes Cuál es la Diferencia entre Videoconferencia y Telepresencia?* (Multimedia) Recuperado el 8 de Abril de 2018, de Multimedia: https://www.multimedia.com.mx/2017/04/28/sabes-cual-es-la-diferencia-entre-videoconferencia-y-telepresencia/

Oracle. (2010). *Modelo de arquitectura del protocolo TCP/IP.* Recuperado el 9 de Mayo de 2018, de Guía de administración del sistema: servicios IP: https://docs.oracle.com/cd/E19957-01/820-2981/ipov-10/

Real Academia Española. (2017). *Telecontrol*. Recuperado el 12 de Abril de 2018, de RAE: http://dle.rae.es/?id=ZLX5yDd

Rouse, M. (29 de Diciembre de 2016). *Red de área extensa (WAN)*. (SearhDataCenter en Español.com) Recuperado el 25 de Mayo de 2018, de SearchDataCenter en Español: https://searchdatacenter.techtarget.com/es/definicion/Red-de-area-extensa-WAN

Singh, P. (5 de Abril de 2016). *NodeMCU Pinout*. (IOT Bytes) Recuperado el Septiembre de 2018, de https://iotbytes.wordpress.com/nodemcu-pinout/

Techno Trends. (s.f.). *Qué es la telepresencia*. Recuperado el 12 de Abril de 2018, de Techno Trends: http://www.salasdetelepresencia.com/que-es-la-telepresencia.html

Texas Instruments. (Enero de 2016). *L293x Quadruple Half-H Drivers.* Recuperado el 21 de Julio de 2018, de Texas Instruments: http://www.ti.com/lit/ds/symlink/l293.pdf

Textos científicos. (2 de Octubre de 2006). *TCP/IP*. (Textos cientificos.com) Recuperado el 2 de Mayo de 2018, de Textos cientificos: https://www.textoscientificos.com/redes/tcp-ip

Textos científicos. (2 de Octubre de 2016). *Estudios por capas de modelo de arquitectura TCP/IP*. (Textos científicos) Recuperado el 9 de Mayo de 2018, de Textos científicos: https://www.textoscientificos.com/redes/tcp-ip/capas-arquitectura-tcp-ip

TicWeb. (2015). *El paso de la videoconferencia a la telepresencia*. Recuperado el 8 de Abril de 2018, de TicWeb.es : https://www.ticweb.es/el-paso-de-la-videoconferencia-a-la-telepresencia/

Universidad de Santiago de Chile. (s.f.). *Elementos constitutivos de un robot industrial*. Recuperado el 4 de Abril de 2018, de U de Santiago Virtual: http://www.udesantiagovirtual.cl/moodle2/mod/book/view.php?id=24908&chapterid=212

Valencia, G., & Calixto, J. (s.f.). *Modelo TCP/IP.* Recuperado el 2 de Mayo de 2018, de Cisco Systems Networking Academy: http://giret.ufps.edu.co/cisco/docs/material/ccna1_cap11b.pdf

Tabla de ilustraciones.

Apéndices.

A. Proveedores de Internet y su importancia a la hora de montar un servidor.

Es muy importante realizar una adecuada selección de nuestro proveedor de Internet si deseamos montar un servidor. En México existen varias empresas que se dedican a ofrecer este servicio en toda la república, algunas de esas empresas son:

- Axtel.
- Izzi.
- Megacable.
- Telnor.
- Telmex.
- TotalPlay.

No solo debemos fijarnos en características como el tipo y la velocidad de conexión, sino que debemos prestar atención al tipo de dirección IP que nos ofrecen pues de esto dependerá si podremos conectarnos a nuestro servidor de manera remota o no.

En nuestro caso contamos con el servicio proporcionado por TotalPlay el cual ofrece una buena velocidad de conexión y estabilidad, sin embargo, presenta el problema de ofrecer direcciones IP públicas dinámicas y con NAT.

Una IP pública es aquella que se le asigna a nuestro router y permite que este sea identificado desde el exterior, cuando esta dirección es dinámica significa que estará cambiando cada cierto tiempo. Por otro lado, la NAT (Network Address Translation, Traducción de Direcciones de Red) fue pensada debido a la escasez de direcciones IP derivadas del IPv4, dicha escasez ocasiona que las direcciones IP disponibles no alcancen para asignar una diferente a cada dispositivo que requiere conectarse a Internet, por lo tanto, la NAT permite que una red de ordenadores o dispositivos utilicen un rango de direcciones IP privadas y se conecten a la red utilizando una única dirección IP pública (la de nuestro router). Esta NAT puede funcionar de diversas maneras, pero en el caso de TotalPlay funciona de manera que visto desde fuera de nuestra red todos los dispositivos conectados poseen la misma dirección IP, esto no significa que todos los dispositivos tengan realmente la misma dirección IP sino que desde el exterior da esa impresión, a este funcionamiento se le conoce como NAT dinámica.

Lo anterior ocasiona dos problemas, el primero se relaciona con la IP pública dinámica que al estar cambiando cada cierto tiempo deberá de verificarse constantemente para asegurar que estamos intentando conectar con la IP correcta. El segundo tiene que ver con la NAT dinámica y es sin duda el más complicado de resolver pues al mostrar prácticamente la misma dirección IP en todos los

dispositivos, una petición de conexión a la IP de nuestro servidor realizada desde el exterior de la red no podrá ser redirigida al dispositivo o maquina correspondiente ocasionando fallas de conexión.

Una solución al problema de la NAT es la llamada “apertura de puertos”, no obstante, TotalPlay bloqueara todo intento por abrir puertos para permitir la comunicación del servidor con el exterior de la red.

La empresa Telmex sí ofrece la posibilidad de comunicarnos desde el exterior pues el funcionamiento de su NAT es estático, es decir, las IP privadas son diferenciables desde el exterior.

Por todo lo anterior es ampliamente recomendable no contratar el servicio de TotalPlay o de Izzi si se desea crear un servidor con comunicación con el exterior, en dado caso se recomienda contratar el servicio ofrecido por Telmex.

B. Diagrama del circuito para el control de motores.

A continuación, se muestra el esquemático realizado en el software EAGLE para la fabricación de dicho circuito.

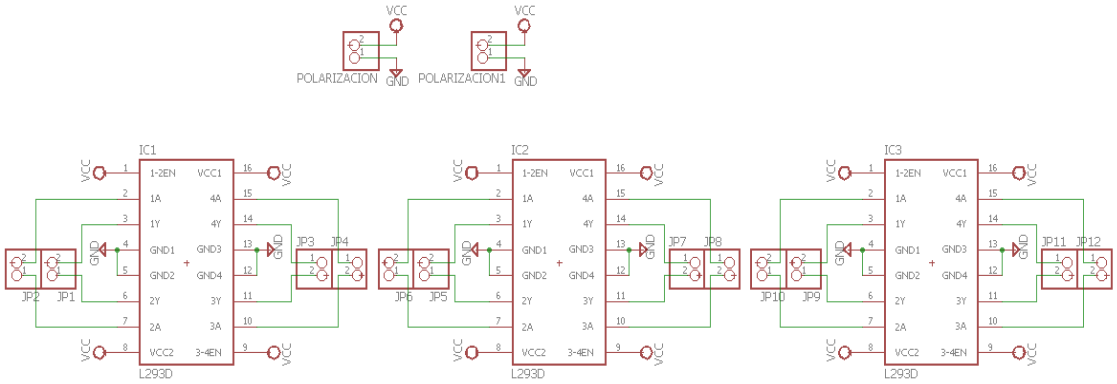


Figura 132: Esquemático para la fabricación del circuito controlador de motores. Elaboración propia.

Para la comodidad del lector se anexará el aspecto del board realizado igualmente en EAGLE, así como la imagen que debe ser impresa sobre papel transfer o albanene si se desea fabricar el circuito.

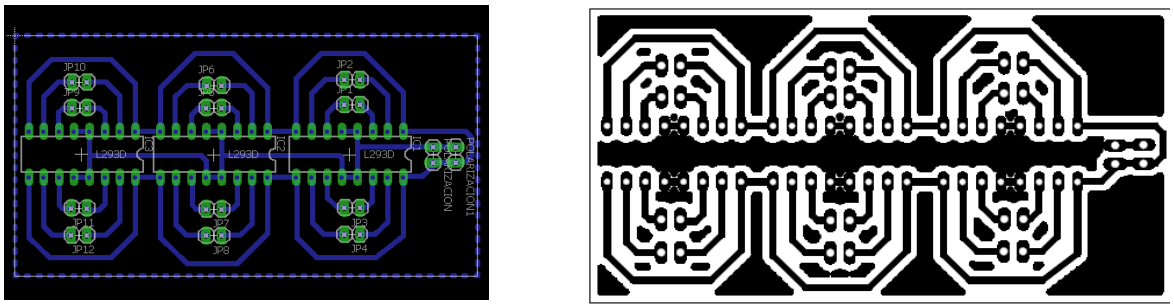


Figura 133: Board e imagen para impreso en tamaño real. Elaboración propia.

C. Ethernet v1.

Anteriormente algunas partes del código fueron mostradas mediante impresiones de pantalla, pero ahora será transcrito para que el lector pueda verificar su funcionamiento. Lo mismo se realizará con el resto de códigos.

```
//PROGRAMA PARA CONTROL DE BRAZO ROBOTICO A TRAVES DE INTERNET
//CREADO POR ISMAEL ANGEL SALAZAR RUIZ ING. ELÉCTRICA-ELECTRÓNICA FES ARAGON UNAM
//AÑO: 2018
//PROYECTO PARA TITULACIÓN
//*****************************************************************************************************************************************
***
//Se declaran las librerias a utilizar para obtener comunicación con el ETHERNET SHIELD modelo W5100
#include <SPI.h>      // Incluimos la biblioteca SPI
#include <Ethernet.h> // Incluimos la biblioteca Ethernet

//Declaramos los pines correspondientes a cada motor y a cada sentido de giro

  int baseIzq = 0;
  int baseDer = 1;
  int hombroUp = 2;
  int hombroDown = 3;
  int codoUp = 4;
  int codoDown = 5;
  int munecaUp = 6;
  int munecaDown = 7;
  int pinzaAbrir = 8;
  int pinzaCerrar = 9;

//Se declaran las configuraciones necesarias para la comunicación con Internet como la direcciones MAC e IP. También del
puerto 80

  byte mac[]={0xDE,0xAD,0xBE,0xEF,0xFE,0xED}; //MAC
  IPAddress ip(192,168,100,177); //IP
  EthernetServer servidor(80);

  String readString=String(30);

void setup()
{
  Ethernet.begin(mac, ip); //Inicializamos con las direcciones asignadas
  servidor.begin();

//configuración de los pines

  pinMode (baseIzq,OUTPUT);
  pinMode (baseDer,OUTPUT);
  pinMode (hombroUp,OUTPUT);
  pinMode (hombroDown,OUTPUT);
  pinMode (codoUp,OUTPUT);
  pinMode (codoDown,OUTPUT);
  pinMode (munecaUp,OUTPUT);
  pinMode (munecaDown,OUTPUT);
  pinMode (pinzaAbrir,OUTPUT);
  pinMode (pinzaCerrar,OUTPUT);

//Colocamos los pines en 0 para iniciar

  digitalWrite (baseIzq,LOW);
  digitalWrite (baseDer,LOW);
  digitalWrite (hombroUp,LOW);
  digitalWrite (hombroDown,LOW);
  digitalWrite (codoUp,LOW);
```

```
  digitalWrite (codoDown,LOW);
  digitalWrite (munecaUp,LOW);
  digitalWrite (munecaDown,LOW);
  digitalWrite (pinzaAbrir,LOW);
  digitalWrite (pinzaCerrar,LOW);
}

void loop()
{
  EthernetClient cliente= servidor.available();

  if(cliente)
  {
    boolean lineaenblanco=true;
    while(cliente.connected())//Cliente conectado
    {
      if(cliente.available())
      {
        char c=cliente.read();
        if(readString.length()<30)//Leemos petición HTTP caracter a caracter
        {
          readString.concat(c); //Almacenar los caracteres en la variable readString
        }
        if(c=='\n' && lineaenblanco)//Si la petición HTTP ha finalizado
        {
          int BASE = readString.indexOf("BASE=");
          int HOMBRO = readString.indexOf("HOMBRO=");
          int CODO = readString.indexOf("CODO=");
          int MUNECA = readString.indexOf("MUNECA=");
          int PINZA = readString.indexOf("PINZA=");

//Movimiento de la base

          if(readString.substring(BASE,BASE+14)=="BASE=IZQUIERDA")
          {
            digitalWrite (baseIzq,HIGH);
            digitalWrite (baseDer,LOW);
            delay(500);
            digitalWrite (baseIzq,LOW);
            digitalWrite (baseDer,LOW);
            delay(1);
          }
          else if (readString.substring(BASE,BASE+12)=="BASE=DERECHA")
          {
            digitalWrite (baseIzq,LOW);
            digitalWrite (baseDer,HIGH);
            delay(500);
            digitalWrite (baseIzq,LOW);
            digitalWrite (baseDer,LOW);
            delay(1);
          }

//Movimiento del hombro

         else if (readString.substring(HOMBRO,HOMBRO+13)=="HOMBRO=ARRIBA")
          {
            digitalWrite (hombroUp,HIGH);
            digitalWrite (hombroDown,LOW);
            delay(500);
            digitalWrite (hombroUp,LOW);
            digitalWrite (hombroDown,LOW);
            delay(1);
          }
         else if (readString.substring(HOMBRO,HOMBRO+12)=="HOMBRO=ABAJO")
          {
```

```
            digitalWrite (hombroUp,LOW);
            digitalWrite (hombroDown,HIGH);
            delay(500);
            digitalWrite (hombroUp,LOW);
            digitalWrite (hombroDown,LOW);
            delay(1);
          }
//Movimiento del codo

          else if (readString.substring(CODO,CODO+11)=="CODO=ARRIBA")
          {
            digitalWrite (codoUp,HIGH);
            digitalWrite (codoDown,LOW);
            delay(500);
            digitalWrite (codoUp,LOW);
            digitalWrite (codoDown,LOW);
            delay(1);
          }
          else if (readString.substring(CODO,CODO+10)=="CODO=ABAJO")
          {
            digitalWrite (codoUp,LOW);
            digitalWrite (codoDown,HIGH);
            delay(500);
            digitalWrite (codoUp,LOW);
            digitalWrite (codoDown,LOW);
            delay(1);
          }

//Movimiento de la muñeca

          else if (readString.substring(MUNECA,MUNECA+13)=="MUNECA=ARRIBA")
          {
            digitalWrite (munecaUp,HIGH);
            digitalWrite (munecaDown,LOW);
            delay(500);;
            digitalWrite (munecaUp,LOW);
            digitalWrite (munecaDown,LOW);
            delay(1);
          }
          else if (readString.substring(MUNECA,MUNECA+12)=="MUNECA=ABAJO")
          {
            digitalWrite (munecaUp,LOW);
            digitalWrite (munecaDown,HIGH);
            delay(500);;
            digitalWrite (munecaUp,LOW);
            digitalWrite (munecaDown,LOW);
            delay(1);
          }

//Movimiento de la pinza

          else if (readString.substring(PINZA,PINZA+11)=="PINZA=ABRIR")
          {
            digitalWrite (pinzaAbrir,HIGH);
            digitalWrite (pinzaCerrar,LOW);
            delay(500);
            digitalWrite (pinzaAbrir,LOW);
            digitalWrite (pinzaCerrar,LOW);
            delay(1);
          }
          else if (readString.substring(PINZA,PINZA+12)=="PINZA=CERRAR")
          {
            digitalWrite (pinzaAbrir,LOW);
            digitalWrite (pinzaCerrar,HIGH);
            delay(500);
```

```
          digitalWrite (pinzaAbrir,LOW);
          digitalWrite (pinzaCerrar,LOW);
          delay(1);
        }

        //Cabecera HTTP estándar
        cliente.println("HTTP/1.1 200 OK");
        cliente.println("Content-Type: text/html;charset=utf-8");
        cliente.println();
        //Página Web en HTML
        cliente.println("<html>");
        cliente.println("<head>");
        cliente.println("<title>PANEL DE CONTROL BRAZO ROBÓTICO</title>");
        cliente.println("</head>");
        cliente.println("<body width=100% height=100%>");
        cliente.println("<center>");
        cliente.println("<h1>FES ARAGÓN</h1>");
        cliente.println("<h1>PANEL DE CONTROL BRAZO ROBÓTICO</h1>");
        cliente.print("<br><br>");
        cliente.print("Base: ");
        cliente.print("<br>");
        cliente.println(F("<input type=submit value=← style=width:200px;height:75px
onClick=location.href='./?BASE=IZQUIERDA\'>"));
        cliente.println(F("<input type=submit value=→ style=width:200px;height:75px
onClick=location.href='./?BASE=DERECHA\'>"));
        cliente.print("<br>");
        cliente.print("Hombro: ");
        cliente.print("<br>");
        cliente.println(F("<input type=submit value=↑ style=width:200px;height:75px
onClick=location.href='./?HOMBRO=ARRIBA\'>"));
        cliente.println(F("<input type=submit value=↓ style=width:200px;height:75px
onClick=location.href='./?HOMBRO=ABAJO\'>"));
         cliente.print("<br>");
        cliente.print("Codo: ");
        cliente.print("<br>");
        cliente.println(F("<input type=submit value=↑ style=width:200px;height:75px
onClick=location.href='./?CODO=ARRIBA\'>"));
        cliente.println(F("<input type=submit value=↓ style=width:200px;height:75px
onClick=location.href='./?CODO=ABAJO\'>"));
        cliente.print("<br>");
        cliente.print("Muñeca: ");
        cliente.print("<br>");
        cliente.println(F("<input type=submit value=↑ style=width:200px;height:75px
onClick=location.href='./?MUNECA=ARRIBA\'>"));
        cliente.println(F("<input type=submit value=↓ style=width:200px;height:75px
onClick=location.href='./?MUNECA=ABAJO\'>"));
         cliente.print("<br>");
        cliente.print("Pinza: ");
        cliente.print("<br>");
        cliente.println(F("<input type=submit value=ABRIR style=width:200px;height:75px
onClick=location.href='./?PINZA=ABRIR\'>"));
        cliente.println(F("<input type=submit value=CERRAR style=width:200px;height:75px
onClick=location.href='./?PINZA=CERRAR\'>"));
        cliente.println("</center>");
        cliente.println("</body>");
        cliente.println("</html>");
        cliente.stop();//Cierro conexión con el clientex
        readString="";
      }
    }
  }
 }
}
```

D. Ethernet v2.

```
//PROGRAMA PARA CONTROL DE BRAZO ROBOTICO A TRAVES DE INTERNET
//CREADO POR ISMAEL ANGEL SALAZAR RUIZ ING. ELÉCTRICA-ELECTRÓNICA FES ARAGON UNAM
//AÑO: 2018
//PROYECTO PARA TITULACIÓN
//*****************************************************************************************************************************************
***
//Se declaran las librerias a utilizar para obtener comunicación con el ETHERNET SHIELD modelo W5100
#include <SPI.h>      // Incluimos la biblioteca SPI
#include <Ethernet.h> // Incluimos la biblioteca Ethernet

//Declaramos los pines correspondientes a cada motor y a cada sentido de giro

  int baseIzq = 0;
  int baseDer = 1;
  int hombroUp = 2;
  int hombroDown = 3;
  int codoUp = 4;
  int codoDown = 5;
  int munecaUp = 6;
  int munecaDown = 7;
  int pinzaAbrir = 8;
  int pinzaCerrar = 9;

//Se declaran las configuraciones necesarias para la comunicación con Internet como la direcciones MAC e IP. También del
puerto 80

  byte mac[]={0xDE,0xAD,0xBE,0xEF,0xFE,0xED}; //MAC
  IPAddress ip(192,168,100,177); //IP
  EthernetServer servidor(80);

  String readString=String(30);

void setup()
{
  Ethernet.begin(mac, ip); //Inicializamos con las direcciones asignadas
  servidor.begin();

//configuración de los pines

  pinMode (baseIzq,OUTPUT);
  pinMode (baseDer,OUTPUT);
  pinMode (hombroUp,OUTPUT);
  pinMode (hombroDown,OUTPUT);
  pinMode (codoUp,OUTPUT);
  pinMode (codoDown,OUTPUT);
  pinMode (munecaUp,OUTPUT);
  pinMode (munecaDown,OUTPUT);
  pinMode (pinzaAbrir,OUTPUT);
  pinMode (pinzaCerrar,OUTPUT);

//Colocamos los pines en 0 para iniciar

  digitalWrite (baseIzq,LOW);
  digitalWrite (baseDer,LOW);
  digitalWrite (hombroUp,LOW);
  digitalWrite (hombroDown,LOW);
  digitalWrite (codoUp,LOW);
  digitalWrite (codoDown,LOW);
  digitalWrite (munecaUp,LOW);
  digitalWrite (munecaDown,LOW);
  digitalWrite (pinzaAbrir,LOW);
  digitalWrite (pinzaCerrar,LOW);
```

```
}

void loop()
{
  EthernetClient cliente= servidor.available();

  if(cliente)
  {
    boolean lineaenblanco=true;
    while(cliente.connected())//Cliente conectado
    {
      if(cliente.available())
      {
        char c=cliente.read();
        if(readString.length()<30)//Leemos petición HTTP caracter a caracter
        {
          readString.concat(c); //Almacenar los caracteres en la variable readString
        }
        if(c=='\n' && lineaenblanco)//Si la petición HTTP ha finalizado
        {
          int BASE = readString.indexOf("BASE=");
          int HOMBRO = readString.indexOf("HOMBRO=");
          int CODO = readString.indexOf("CODO=");
          int MUNECA = readString.indexOf("MUNECA=");
          int PINZA = readString.indexOf("PINZA=");
          int ESCLAVO = readString.indexOf("ESCLAVO=");

//Movimiento de la base

          if(readString.substring(BASE,BASE+14)=="BASE=IZQUIERDA")
          {
             digitalWrite (baseIzq,HIGH);
             digitalWrite (baseDer,LOW);
          }
          else if (readString.substring(BASE,BASE+12)=="BASE=DERECHA")
          {
             digitalWrite (baseIzq,LOW);
             digitalWrite (baseDer,HIGH);
          }

//Movimiento del hombro

         else if (readString.substring(HOMBRO,HOMBRO+13)=="HOMBRO=ARRIBA")
          {
             digitalWrite (hombroUp,HIGH);
             digitalWrite (hombroDown,LOW);
          }
         else if (readString.substring(HOMBRO,HOMBRO+12)=="HOMBRO=ABAJO")
          {
             digitalWrite (hombroUp,LOW);
             digitalWrite (hombroDown,HIGH);
          }
//Movimiento del codo

          else if (readString.substring(CODO,CODO+11)=="CODO=ARRIBA")
          {
             digitalWrite (codoUp,HIGH);
             digitalWrite (codoDown,LOW);
          }
          else if (readString.substring(CODO,CODO+10)=="CODO=ABAJO")
          {
             digitalWrite (codoUp,LOW);
             digitalWrite (codoDown,HIGH);
          }
```

```
//Movimiento de la muñeca

        else if (readString.substring(MUNECA,MUNECA+13)=="MUNECA=ARRIBA")
        {
           digitalWrite (munecaUp,HIGH);
           digitalWrite (munecaDown,LOW);
        }
        else if (readString.substring(MUNECA,MUNECA+12)=="MUNECA=ABAJO")
        {
           digitalWrite (munecaUp,LOW);
           digitalWrite (munecaDown,HIGH);
        }

//Movimiento de la pinza

        else if (readString.substring(PINZA,PINZA+11)=="PINZA=ABRIR")
        {
           digitalWrite (pinzaAbrir,HIGH);
           digitalWrite (pinzaCerrar,LOW);
        }
        else if (readString.substring(PINZA,PINZA+12)=="PINZA=CERRAR")
        {
           digitalWrite (baseIzq,LOW);
           digitalWrite (baseDer,LOW);
           digitalWrite (hombroUp,LOW);
           digitalWrite (hombroDown,LOW);
           digitalWrite (codoUp,LOW);
           digitalWrite (codoDown,LOW);
           digitalWrite (munecaUp,LOW);
           digitalWrite (munecaDown,LOW);
           digitalWrite (pinzaAbrir,LOW);
           digitalWrite (pinzaCerrar,HIGH);
        }

//Esclavo sin movimiento

        else if (readString.substring(ESCLAVO,ESCLAVO+15)=="ESCLAVO=APAGADO")
        {
           digitalWrite (baseIzq,LOW);
           digitalWrite (baseDer,LOW);
           digitalWrite (hombroUp,LOW);
           digitalWrite (hombroDown,LOW);
           digitalWrite (codoUp,LOW);
           digitalWrite (codoDown,LOW);
           digitalWrite (munecaUp,LOW);
           digitalWrite (munecaDown,LOW);
           digitalWrite (pinzaAbrir,LOW);
           digitalWrite (pinzaCerrar,LOW);
         }

         //Cabecera HTTP estándar
         cliente.println("HTTP/1.1 200 OK");
         cliente.println("Content-Type: text/html;charset=utf-8");
         cliente.println();
         //Página Web en HTML
         cliente.println("<html>");
         cliente.println("<head>");
         cliente.println("<title>PANEL DE CONTROL BRAZO ROBÓTICO</title>");
         cliente.println("</head>");
         cliente.println("<body width=100% height=100%>");
         cliente.println("<center>");
         cliente.println("<h1>FES ARAGÓN</h1>");
         cliente.println("<h1>PANEL DE CONTROL BRAZO ROBÓTICO</h1>");
         cliente.print("<br><br>");
```

```
        cliente.println(F("<input type='button' value=\"Paro de emergencia\" style=width:200px;height:75px
onClick=\"location.href='/?ESCLAVO=APAGADO';\">"));
        cliente.print("<br><br>");
        cliente.print("Base: ");
        cliente.print("<br>");
        cliente.println(F("<input type='button' value=\"←\" style=width:200px;height:75px
onpointerdown=\"location.href='/?BASE=IZQUIERDA';\" onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
        cliente.println(F("<input type='button' value=\"→\" style=width:200px;height:75px
onpointerdown=\"location.href='/?BASE=DERECHA';\" onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
        cliente.print("<br>");
        cliente.print("Hombro: ");
        cliente.print("<br>");
        cliente.println(F("<input type='button' value=\"↑\" style=width:200px;height:75px
onpointerdown=\"location.href='/?HOMBRO=ARRIBA';\" onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
        cliente.println(F("<input type='button' value=\"↓\" style=width:200px;height:75px
onpointerdown=\"location.href='/?HOMBRO=ABAJO';\" onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
         cliente.print("<br>");
        cliente.print("Codo: ");
        cliente.print("<br>");
        cliente.println(F("<input type='button' value=\"↑\" style=width:200px;height:75px
onpointerdown=\"location.href='/?CODO=ARRIBA';\" onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
        cliente.println(F("<input type='button' value=\"↓\" style=width:200px;height:75px
onpointerdown=\"location.href='/?CODO=ABAJO';\" onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
        cliente.print("<br>");
        cliente.print("Muñeca: ");
        cliente.print("<br>");
        cliente.println(F("<input type='button' value=\"↑\" style=width:200px;height:75px
onpointerdown=\"location.href='/?MUNECA=ARRIBA';\" onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
        cliente.println(F("<input type='button' value=\"↓\" style=width:200px;height:75px
onpointerdown=\"location.href='/?MUNECA=ABAJO';\" onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
         cliente.print("<br>");
        cliente.print("Pinza: ");
        cliente.print("<br>");
        cliente.println(F("<input type='button' value=\"ABRIR\" style=width:200px;height:75px
onpointerdown=\"location.href='/?PINZA=ABRIR';\" onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
        cliente.println(F("<input type='button' value=\"CERRAR\" style=width:200px;height:75px
onpointerdown=\"location.href='/?PINZA=CERRAR';\" onpointerup=\"location.href='/?ESCLAVO=APAGADO';\">"));
        cliente.println("</center>");
        cliente.println("</body>");
        cliente.println("</html>");
        cliente.stop();//Cierro conexión con el cliente
        readString="";
      }
    }
  }
 }
}
```

E. Wi-Fi v1.

```
//PROGRAMA PARA CONTROL DE BRAZO ROBOTICO A TRAVES DE INTERNET
//CREADO POR ISMAEL ANGEL SALAZAR RUIZ ING. ELÉCTRICA-ELECTRÓNICA FES ARAGON UNAM
//AÑO: 2018
//PROYECTO PARA TITULACIÓN
//*************************************************************************************************************************************************
***

#include <ESP8266WiFi.h>
#include <ESP8266WiFiMulti.h>
#include <ESP8266WebServer.h>

ESP8266WiFiMulti WiFiMulti;

//Declaramos los pines correspondientes a cada motor y a cada sentido de giro

  int baseIzq = 16;
  int baseDer = 5;
  int hombroUp = 4;
  int hombroDown = 0;
  int codoUp = 2;
  int codoDown = 14;
  int munecaUp = 12;
  int munecaDown = 13;
  int pinzaAbrir = 15;
  int pinzaCerrar = 3;

//puerto
ESP8266WebServer server(80);

void handleRoot() {
     response();
}

void BaseIzq() {
  digitalWrite (baseIzq,HIGH);
  digitalWrite (baseDer,LOW);
  delay(500);
  digitalWrite (baseIzq,LOW);
  digitalWrite (baseDer,LOW);
  delay(1);
  response();
}

void BaseDer() {
  digitalWrite (baseIzq,LOW);
  digitalWrite (baseDer,HIGH);
  delay(500);
  digitalWrite (baseIzq,LOW);
  digitalWrite (baseDer,LOW);
  delay(1);
  response();
}

void HombroArriba() {
  digitalWrite (hombroUp,HIGH);
  digitalWrite (hombroDown,LOW);
  delay(500);
  digitalWrite (hombroUp,LOW);
  digitalWrite (hombroDown,LOW);
  delay(1);
  response();
}
```

```
void HombroAbajo() {
  digitalWrite (hombroUp,LOW);
  digitalWrite (hombroDown,HIGH);
  delay(500);
  digitalWrite (hombroUp,LOW);
  digitalWrite (hombroDown,LOW);
  delay(1);
  response();
}

void CodoArriba() {
  digitalWrite (codoUp,HIGH);
  digitalWrite (codoDown,LOW);
  delay(500);
  digitalWrite (codoUp,LOW);
  digitalWrite (codoDown,LOW);
  delay(1);
  response();
}

void CodoAbajo() {
  digitalWrite (codoUp,LOW);
  digitalWrite (codoDown,HIGH);
  delay(500);
  digitalWrite (codoUp,LOW);
  digitalWrite (codoDown,LOW);
  delay(1);
  response();
}

void MunecaArriba() {
  digitalWrite (munecaUp,HIGH);
  digitalWrite (munecaDown,LOW);
  delay(500);;
  digitalWrite (munecaUp,LOW);
  digitalWrite (munecaDown,LOW);
  delay(1);
  response();
}

void MunecaAbajo() {
  digitalWrite (munecaUp,LOW);
  digitalWrite (munecaDown,HIGH);
  delay(500);;
  digitalWrite (munecaUp,LOW);
  digitalWrite (munecaDown,LOW);
  delay(1);
  response();
}

void PinzaAbrir() {
  digitalWrite (pinzaAbrir,HIGH);
  digitalWrite (pinzaCerrar,LOW);
  delay(500);
  digitalWrite (pinzaAbrir,LOW);
  digitalWrite (pinzaCerrar,LOW);
  delay(1);
  response();
}

void PinzaCerrar() {
  digitalWrite (pinzaAbrir,LOW);
  digitalWrite (pinzaCerrar,HIGH);
  delay(500);
```

```
  digitalWrite (pinzaAbrir,LOW);
  digitalWrite (pinzaCerrar,LOW);
  delay(1);
  response();
}

const String HtmlHtml = "<html><head><title>PANEL DE CONTROL BRAZO ROBÓTICO</title></head>";
const String HtmlHtmlClose = "</html>";
const String HtmlTitle = "<h1><CENTER>FES ARAGÓN</CENTER></h1>";
const String Htmlbody = "<body width=100% height=100%>";
const String HtmlbodyClose = "</body>";
const String HtmlTitulo = "<h1><CENTER>PANEL DE CONTROL DE BRAZO ROBÓTICO</h1><br>";

const String HtmlBaseIzquierda = "<CENTER>Base:<br><input type=submit value=← style=width:200px;height:75px
onClick=location.href='BASE=IZQUIERDA\'>";
const String HtmlBaseDerecha = "<input type=submit value=→ style=width:200px;height:75px
onClick=location.href='BASE=DERECHA\'><br>";

const String HtmlHombroArriba = "<CENTER>Hombro:<br><input type=submit value=↑ style=width:200px;height:75px
onClick=location.href='HOMBRO=ARRIBA\'>";
const String HtmlHombroAbajo = "<input type=submit value=↓ style=width:200px;height:75px
onClick=location.href='HOMBRO=ABAJO\'><br>";

const String HtmlCodoArriba = "<CENTER>Codo:<br><input type=submit value=↑ style=width:200px;height:75px
onClick=location.href='CODO=ARRIBA\'>";
const String HtmlCodoAbajo = "<input type=submit value=↓ style=width:200px;height:75px
onClick=location.href='CODO=ABAJO\'><br>";

const String HtmlMunecaArriba = "<CENTER>Muñeca:<br><input type=submit value=↑ style=width:200px;height:75px
onClick=location.href='MUNECA=ARRIBA\'>";
const String HtmlMunecaAbajo = "<input type=submit value=↓ style=width:200px;height:75px
onClick=location.href='MUNECA=ABAJO\'><br>";

const String HtmlPinzaAbrir = "<CENTER>Pinza:<br><input type=submit value=ABRIR style=width:200px;height:75px
onClick=location.href='PINZA=ABRIR\'>";
const String HtmlPinzaCerrar = "<input type=submit value=CERRAR style=width:200px;height:75px
onClick=location.href='PINZA=CERRAR\'>";

void response(){
  String
htmlRes=HtmlHtml+HtmlTitle+HtmlTitulo+Htmlbody+HtmlbodyClose+HtmlBaseIzquierda+HtmlBaseDerecha+HtmlHombroAr
riba+HtmlHombroAbajo+HtmlCodoArriba+HtmlCodoAbajo+HtmlMunecaArriba+HtmlMunecaAbajo+HtmlPinzaAbrir+HtmlPinz
aCerrar;

 server.send(200, "text/html;charset=utf-8", htmlRes);
}

void setup() {

//configuración de los pines

  pinMode (baseIzq,OUTPUT);
  pinMode (baseDer,OUTPUT);
  pinMode (hombroUp,OUTPUT);
  pinMode (hombroDown,OUTPUT);
  pinMode (codoUp,OUTPUT);
  pinMode (codoDown,OUTPUT);
  pinMode (munecaUp,OUTPUT);
  pinMode (munecaDown,OUTPUT);
  pinMode (pinzaAbrir,OUTPUT);
  pinMode (pinzaCerrar,OUTPUT);

//Colocamos los pines en 0 para iniciar

  digitalWrite (baseIzq,LOW);
```

```
  digitalWrite (baseDer,LOW);
  digitalWrite (hombroUp,LOW);
  digitalWrite (hombroDown,LOW);
  digitalWrite (codoUp,LOW);
  digitalWrite (codoDown,LOW);
  digitalWrite (munecaUp,LOW);
  digitalWrite (munecaDown,LOW);
  digitalWrite (pinzaAbrir,LOW);
  digitalWrite (pinzaCerrar,LOW);
  Serial.begin(115200);
  delay(10);

 WiFiMulti.addAP("Totalplay-874B", "EAF4874B");

   Serial.println();
   Serial.println();
   Serial.print("Conectando con WiFi ");

   while(WiFiMulti.run() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
   }

  Serial.println("");
  Serial.println("WiFi conexion exitosa");
  Serial.println("direccion IP: ");
  Serial.println(WiFi.localIP());
  delay(1000);
  Serial.begin(115200);
  Serial.println();
  Serial.print("Configurando access point...");

  IPAddress myIP = WiFi.softAPIP();
  server.on("/", handleRoot);
  server.on("/BASE=IZQUIERDA", BaseIzq);
  server.on("/BASE=DERECHA", BaseDer);
  server.on("/HOMBRO=ARRIBA", HombroArriba);
  server.on("/HOMBRO=ABAJO", HombroAbajo);
  server.on("/CODO=ARRIBA", CodoArriba);
  server.on("/CODO=ABAJO", CodoAbajo);
  server.on("/MUNECA=ARRIBA", MunecaArriba);
  server.on("/MUNECA=ABAJO", MunecaAbajo);
  server.on("/PINZA=ABRIR", PinzaAbrir);
  server.on("/PINZA=CERRAR", PinzaCerrar);

  server.begin();
  Serial.println("Servicio HTTP Iniciado ");

}

void loop() {

  server.handleClient();

}
```

F. Wi-Fi v2.

```
//PROGRAMA PARA CONTROL DE BRAZO ROBOTICO A TRAVES DE INTERNET
//CREADO POR ISMAEL ANGEL SALAZAR RUIZ ING. ELÉCTRICA-ELECTRÓNICA FES ARAGON UNAM
//AÑO: 2018
//PROYECTO PARA TITULACIÓN
//*************************************************************************************************************************************************
***

#include <ESP8266WiFi.h>
#include <ESP8266WiFiMulti.h>
#include <ESP8266WebServer.h>

ESP8266WiFiMulti WiFiMulti;

//Declaramos los pines correspondientes a cada motor y a cada sentido de giro

  int baseIzq = 16;
  int baseDer = 5;
  int hombroUp = 4;
  int hombroDown = 0;
  int codoUp = 2;
  int codoDown = 14;
  int munecaUp = 12;
  int munecaDown = 13;
  int pinzaAbrir = 15;
  int pinzaCerrar = 3;

//puerto
ESP8266WebServer server(80);

void handleRoot() {
     response();
}

void BaseIzq() {
  digitalWrite (baseIzq,HIGH);
  digitalWrite (baseDer,LOW);
  response();
}

void BaseDer() {
  digitalWrite (baseIzq,LOW);
  digitalWrite (baseDer,HIGH);
  response();
}

void HombroArriba() {
  digitalWrite (hombroUp,HIGH);
  digitalWrite (hombroDown,LOW);
  response();
}

void HombroAbajo() {
  digitalWrite (hombroUp,LOW);
  digitalWrite (hombroDown,HIGH);
  response();
}

void CodoArriba() {
  digitalWrite (codoUp,HIGH);
  digitalWrite (codoDown,LOW);
  response();
}
```

```
void CodoAbajo() {
  digitalWrite (codoUp,LOW);
  digitalWrite (codoDown,HIGH);
  response();
}

void MunecaArriba() {
  digitalWrite (munecaUp,HIGH);
  digitalWrite (munecaDown,LOW);
  response();
}

void MunecaAbajo() {
  digitalWrite (munecaUp,LOW);
  digitalWrite (munecaDown,HIGH);
  response();
}

void PinzaAbrir() {
  digitalWrite (pinzaAbrir,HIGH);
  digitalWrite (pinzaCerrar,LOW);
  response();
}

void PinzaCerrar() {
  digitalWrite (pinzaAbrir,LOW);
  digitalWrite (pinzaCerrar,HIGH);
  response();
}

void Off(){
  digitalWrite (baseIzq,LOW);
  digitalWrite (baseDer,LOW);
  digitalWrite (hombroUp,LOW);
  digitalWrite (hombroDown,LOW);
  digitalWrite (codoUp,LOW);
  digitalWrite (codoDown,LOW);
  digitalWrite (munecaUp,LOW);
  digitalWrite (munecaDown,LOW);
  digitalWrite (pinzaAbrir,LOW);
  digitalWrite (pinzaCerrar,LOW);
  response();
}

const String HtmlHtml = "<html><head><title>PANEL DE CONTROL BRAZO ROBÓTICO</title></head>";
const String HtmlHtmlClose = "</html>";
const String HtmlTitle = "<h1><CENTER>FES ARAGÓN</CENTER></h1>";
const String Htmlbody = "<body width=100% height=100%>";
const String HtmlbodyClose = "</body>";
const String HtmlTitulo = "<h1><CENTER>PANEL DE CONTROL DE BRAZO ROBÓTICO</h1><br>";

const String HtmlOff = "<CENTER>Apagado:<br><input type=submit value=Paro de emergencia
style=width:200px;height:75px onClick=location.href='ESCLAVO=APAGADO\'>";

const String HtmlBaseIzquierda = "<CENTER>Base:<br><input type='button' value=← style=width:200px;height:75px
onpointerdown=\"location.href='BASE=IZQUIERDA';\" onpointerup=\"location.href='ESCLAVO=APAGADO';\">";
const String HtmlBaseDerecha = "<input type=submit value=→ style=width:200px;height:75px
onpointerdown=\"location.href='BASE=DERECHA';\" onpointerup=\"location.href='ESCLAVO=APAGADO';\"><br>";

const String HtmlHombroArriba = "<CENTER>Hombro:<br><input type=submit value=↑ style=width:200px;height:75px
onpointerdown=\"location.href='HOMBRO=ARRIBA';\" onpointerup=\"location.href='ESCLAVO=APAGADO';\">";
const String HtmlHombroAbajo = "<input type=submit value=↓ style=width:200px;height:75px
onpointerdown=\"location.href='HOMBRO=ABAJO';\" onpointerup=\"location.href='ESCLAVO=APAGADO';\"><br>";
```

```
const String HtmlCodoArriba = "<CENTER>Codo:<br><input type=submit value=↑ style=width:200px;height:75px
onpointerdown=\"location.href='CODO=ARRIBA';\" onpointerup=\"location.href='ESCLAVO=APAGADO';\">";
const String HtmlCodoAbajo = "<input type=submit value=↓ style=width:200px;height:75px
onpointerdown=\"location.href='CODO=ABAJO';\" onpointerup=\"location.href='ESCLAVO=APAGADO';\"><br>";

const String HtmlMunecaArriba = "<CENTER>Muñeca:<br><input type=submit value=↑ style=width:200px;height:75px
onpointerdown=\"location.href='MUNECA=ARRIBA';\" onpointerup=\"location.href='ESCLAVO=APAGADO';\">";
const String HtmlMunecaAbajo = "<input type=submit value=↓ style=width:200px;height:75px
onpointerdown=\"location.href='MUNECA=ABAJO';\" onpointerup=\"location.href='ESCLAVO=APAGADO';\"><br>";

const String HtmlPinzaAbrir = "<CENTER>Pinza:<br><input type=submit value=ABRIR style=width:200px;height:75px
onpointerdown=\"location.href='PINZA=ABRIR';\" onpointerup=\"location.href='ESCLAVO=APAGADO';\">";
const String HtmlPinzaCerrar = "<input type=submit value=CERRAR style=width:200px;height:75px
onpointerdown=\"location.href='PINZA=CERRAR';\" onpointerup=\"location.href='ESCLAVO=APAGADO';\">";

void response(){
  String
htmlRes=HtmlHtml+HtmlTitle+HtmlTitulo+Htmlbody+HtmlbodyClose+HtmlOff+HtmlBaseIzquierda+HtmlBaseDerecha+HtmlH
ombroArriba+HtmlHombroAbajo+HtmlCodoArriba+HtmlCodoAbajo+HtmlMunecaArriba+HtmlMunecaAbajo+HtmlPinzaAbrir+
HtmlPinzaCerrar;

 server.send(200, "text/html;charset=utf-8", htmlRes);
}

void setup() {

//configuración de los pines

  pinMode (baseIzq,OUTPUT);
  pinMode (baseDer,OUTPUT);
  pinMode (hombroUp,OUTPUT);
  pinMode (hombroDown,OUTPUT);
  pinMode (codoUp,OUTPUT);
  pinMode (codoDown,OUTPUT);
  pinMode (munecaUp,OUTPUT);
  pinMode (munecaDown,OUTPUT);
  pinMode (pinzaAbrir,OUTPUT);
  pinMode (pinzaCerrar,OUTPUT);

//Colocamos los pines en 0 para iniciar

  digitalWrite (baseIzq,LOW);
  digitalWrite (baseDer,LOW);
  digitalWrite (hombroUp,LOW);
  digitalWrite (hombroDown,LOW);
  digitalWrite (codoUp,LOW);
  digitalWrite (codoDown,LOW);
  digitalWrite (munecaUp,LOW);
  digitalWrite (munecaDown,LOW);
  digitalWrite (pinzaAbrir,LOW);
  digitalWrite (pinzaCerrar,LOW);
  Serial.begin(115200);
  delay(10);

 WiFiMulti.addAP("Totalplay-874B", "EAF4874B");

   Serial.println();
   Serial.println();
   Serial.print("Conectando con WiFi ");

   while(WiFiMulti.run() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
   }
```

```
  Serial.println("");
  Serial.println("WiFi conexion exitosa");
  Serial.println("direccion IP: ");
  Serial.println(WiFi.localIP());
  delay(1000);
  Serial.begin(115200);
  Serial.println();
  Serial.print("Configurando access point...");

  IPAddress myIP = WiFi.softAPIP();
  server.on("/", handleRoot);
  server.on("/BASE=IZQUIERDA", BaseIzq);
  server.on("/BASE=DERECHA", BaseDer);
  server.on("/HOMBRO=ARRIBA", HombroArriba);
  server.on("/HOMBRO=ABAJO", HombroAbajo);
  server.on("/CODO=ARRIBA", CodoArriba);
  server.on("/CODO=ABAJO", CodoAbajo);
  server.on("/MUNECA=ARRIBA", MunecaArriba);
  server.on("/MUNECA=ABAJO", MunecaAbajo);
  server.on("/PINZA=ABRIR", PinzaAbrir);
  server.on("/PINZA=CERRAR", PinzaCerrar);
  server.on("/ESCLAVO=APAGADO",Off);

  server.begin();
  Serial.println("Servicio HTTP Iniciado ");

}

void loop() {

  server.handleClient();

}
```