CONTROL DEL BRAZO ROBOT UR3 DE UNIVERSAL ROBOT MEDIANTE IMITACIÓN DE MOVIMIENTO

Oriol Mérida Rubio
Departamento de Ingeniería Electrónica – Escuela Politécnica Superior de Ingeniería de Vilanova y la Geltrú (2023)

RESUMEN

En este proyecto se propone realizar el control de los movimientos de un brazo robótico a partir de los movimientos de un brazo humano. En este caso se trata de, un brazo robot de Universal Robot, más concretamente, el modelo UR3e.
Para captar y detectar los movimientos del brazo humano, se usarán dispositivos IMU (sensores de medición inercial) para captar los movimientos del dispositivo y transformar dicha información para enviarle las órdenes de movimiento al brazo robot.
En este trabajo se va a desarrollar un sistema compuesto, prioritariamente, por sensores inerciales donde, primero de todo, se va a describir y seleccionar todas las herramientas y dispositivos necesarios para realizar dicho proyecto (Visual Studio Code, URSim, UR3 e, BNO055, ATmega4809, NINA-W102).
Por otro lado, se expondrán las diferentes metodologías que se han usado para intentar captar los movimientos del brazo humano, además de la traducción de esa información captada, para que el UR3 e sea capaz de interpretar esa información en movimiento.
Con este punto de partida, se aplican herramientas específicas del grado en Ingeniería Electrónica Industrial y Automática, para así poder elaborar una propuesta válida conceptual que satisfaga los objetivos fijados en el proyecto.

INTRODUCCIÓN

Este proyecto está centrado en la realización de un sistema capaz de leer las posiciones reales de un brazo humano en todo momento y que, acto seguido, un brazo robot (UR3 e) sea capaz de reproducir dichas posiciones. El objetivo principal de esta investigación es conseguir que el brazo robot sea capaz de imitar el brazo humano para poder llegar realizar tareas a distancia sin la necesidad de estar en la misma sala que el robot.
Desde bien pequeño me ha fascinado el mundo de la robótica. A medida que he ido creciendo, académicamente hablando, me ha despertado más interés este campo, es por eso que decidí encaminar mis estudios hacia este mundo y así poder aportar mi conocimiento y motivación a este sector todavía en expansión.

Metodología

En este proyecto hará falta aplicar una metodología relacionada con el estudio y programación de microcontroladores, en específico, aplicada a los robots colaborativos, que permita entender y actuar en base a las especificaciones del dispositivo.

Estructura

Este proyecto está dividido por capítulos, con los que se pretende desarrollar el proyecto.

1. Aspectos generales

1.1. Definición del robot

Si buscas la definición de la robótica encontrarás varias definiciones diferentes, pero una de las primeras que te aparecerán será: “Técnica que se utiliza en el diseño y la construcción de robots y aparatos que realizan operaciones o trabajos, generalmente en instalaciones industriales y en sustitución de la mano de obra humana.”
Pero en mi opinión es un campo que está en constante evolución, ya que es una disciplina que combina diferentes campos, como la informática, la ingeniería, la mecánica, la electrónica y cada vez más la inteligencia artificial. El objetivo primordial es desarrollar máquinas capaces de interactuar con el entorno y realizar tareas de manera autónoma (normalmente repetitivas) o colaborativas con las personas [1].

1.2. Clasificación de los robots

A continuación, se muestra una clasificación de robots según tres criterios diferentes:

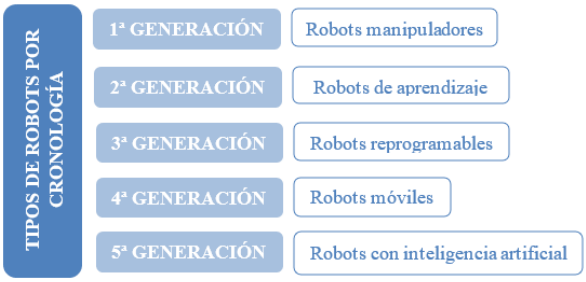

Ilustración 1. *Clasificación de los robots por cronología. Fuente: elaboración propia.*

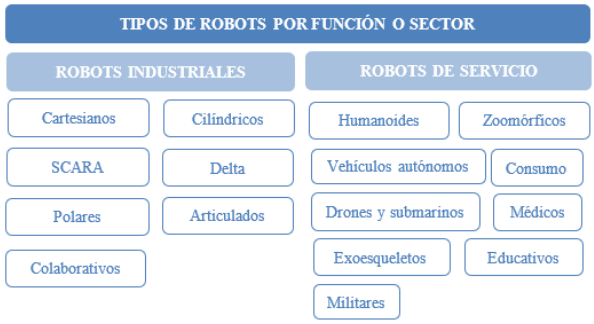

Ilustración 2. *Clasificación de los robots por función o sector. Fuente: elaboración propia.*

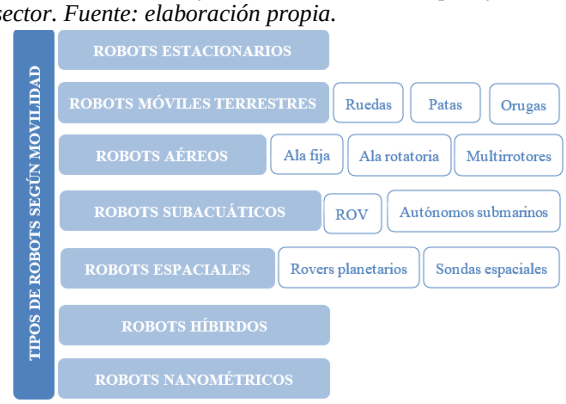

Ilustración 3. *Clasificación de los robots según movilidad. Fuente: elaboración propia.*

1.3. Estructura de los robots industriales

Un manipulador robótico consta de una secuencia de elementos estructurales rígidos, denominados enlaces o eslabones, conectados entre sí mediante juntas o articulaciones, que permiten el movimiento relativo de cada dos eslabones consecutivos [2].

Una articulación puede ser:

- **Lineal** (deslizante, traslacional o prismática), si un eslabón desliza sobre un eje solidario al eslabón anterior.
- **Rotacional**, en caso de que un eslabón gire en torno a un eje solidario al eslabón anterior.

El punto más significativo del elemento terminal se denomina punto terminal (PT). Los elementos terminales pueden dividirse en dos categorías:

- **Pinzas** (*gripper*): se utilizan para tomar un objeto, normalmente la pieza de trabajo, y sujetarlo durante el ciclo de trabajo del robot.
- **Herramientas**: se utilizan como actuador final en aplicaciones en donde se exija al robot realizar alguna operación sobre la pieza de trabajo.

1.4. Configuraciones morfológicas y parámetros característicos de los robots industriales

Según la geometría de su estructura mecánica, un manipulador puede ser [3]: cartesiano, cilíndrico, polar, esférico (o de brazo articulado), mixto, paralelo.
Los principales parámetros que caracterizan a los robots industriales son: el número de grados de libertad (GdL), el espacio de accesibilidad o espacio (volumen) de trabajo, la capacidad de posicionamiento del punto terminal, la capacidad de carga y la velocidad.

1.5. Seguridad para robots colaborativos

Los fabricantes de una amplia variedad de industrias emplean robots colaborativos ("cobots") para aprovechar los beneficios de las funciones de seguridad integradas que permiten que estos robots compartan las tareas con los seres humanos y se adapten de manera flexible a los nuevos requisitos de las nuevas tareas repetitivas. Aunque la seguridad es una característica clave de su diseño, los "cobots" aún requieren evaluaciones de riesgo [4]. La norma de seguridad ISO 10218 y la especificación técnica RIA TS 15066 definen las funciones de seguridad y el rendimiento de los robots.

1.6. Redes de comunicación industrial

En el ámbito de la automatización de procesos industriales, la conectividad de las redes de comunicación entre los diferentes dispositivos que intervienen en el control de estos sistemas, son muy importantes. Esto asegura su correcto funcionamiento, y también facilita el control efectivo de los procesos.
La maquinaria industrial con capacidad de comunicación tiene que funcionar utilizando un protocolo de comunicación ya sea de una forma autónoma o con la manipulación humana. Para un intercambio de datos, es necesario el control de datos y la posibilidad de poder conectar maquinas o dispositivos que sean de fabricantes distintos para que puedan ser útiles en una misma instalación.

1.7. IMU

Una Unidad de Medida Inercial (IMU) es un dispositivo capaz de estimar y reportar estados dinámicos específicos como velocidades angulares y aceleraciones. A partir de estas medidas otros estados dinámicos se pueden inferir, tales como la latitud (roll y pitch), o incrementos en la velocidad y la posición de la plataforma. También puede ser que algunos lleven magnetómetro adicional, para poder medir el campo magnético alrededor del dispositivo.

1.7.1. Sector de uso

En un sistema de navegación inercial, donde los datos sin procesar recopilados por el IMU se procesan más tarde por una computadora que, aplicando diferentes algoritmos, es capaz de estimar la latitud, posición y velocidad de la plataforma, basándose únicamente en los datos aportados por los sensores y resultados de las estimaciones de iteraciones anteriores. Los diferentes algoritmos ejecutados por el sistema son capaces de fusionar las entradas de múltiples sensores del IMU. Esta fusión de sensores, permite detectar datos erróneos o fuera de rango de un determinado sensor, por ejemplo, un magnetómetro afectado por el campo magnético de un objeto cercano. De esta manera, el sistema aísla dicho sensor y sus medidas compensando esta situación con el resto de sensores disponibles y reduciendo el error y la deriva en la estimación. Esto hace que el sistema sea robusto frente a fallos individuales o múltiples de sensores [5].

1.7.2. Componentes

Una IMU se compone normalmente de:

- **Acelerómetros**: miden las fuerzas gravitatorias en un sistema de coordenadas fijo.
- **Giróscopos**: miden la velocidad angular.
- **Magnetómetros**: miden el campo magnético local. Brújula que mide la dirección del campo magnético de la Tierra en 2D.

1.7.3. Cálculo de Euler

La fórmula de Euler es un concepto matemático que relaciona dos conceptos elementales de las matemáticas: los números complejos y la trigonometría. Se escribe como $e^{ix} = \cos(x) + i \cdot sen(x)$, donde e es la base del logaritmo natural, i es la unidad imaginaria (definida como la raíz cuadrada de -1) y x es un número real. Esta ecuación especifica que el número complejo e^{ix} es igual a la suma del número real cos(x) y del producto del número imaginario i por el número real sen(x).

Ángulos de Euler

Los ángulos de Euler son una forma de describir la orientación de un cuerpo rígido en el espacio. Estos ángulos se pueden expresar de diferentes maneras, pero una forma común es mediante los ángulos roll (rotación alrededor del eje X), pitch (rotación alrededor del eje Y) y yaw (rotación alrededor del eje Z) [7].

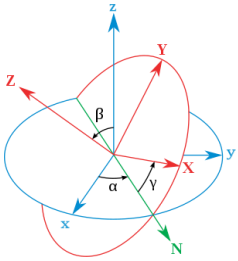

Ilustración 4. *Definición geométrica de los ángulos clásicos de Euler. Fuente: Wikipedia*

2. Universal robot

Universal Robot es una empresa danesa con más de 1.000 empleados, que se dedica a fabricar brazos robóticos colaborativos de 6 ejes, seguros, flexibles e intuitivos para todo tipo de empresas y países. Consiguiendo así ser más versátil a la hora de utilizarlo para la faena que se le requiera.
Hasta la fecha ha sido capaz de vender más de 75.000 "cobots" (son los denominados robots colaborativos) en todo el mundo, que se utilizan en diferentes entornos productivos [5].

2.1. UR3e

El UR3e consiste en un robot compacto de sobremesa, ya que tiene un formato compacto ideal para espacios de trabajo reducidos. Por sus dimensiones, es bastante versátil a la hora de su lugar de trabajo, ya sea en una mesa de trabajo o instalarlo directamente en maquinaria, por esas mismas condiciones es idóneo para aplicaciones ligeras de montaje y atornillado. El UR3e también consta de una consola con la que se hace todo el control del robot. El UR3e también está disponible como un sistema robótico OEM [6] y con una consola de programación con 3 posiciones [7].

2.1.1. Características

Es el robot colaborativo más flexible que tienen, perfecto para tareas de ensamblaje ligeras y bancos de trabajo automatizados. Pesa un total de 11 Kg y su carga útil es de 3 kg solamente, con una rotación de 360° en todos los ejes y una rotación infinita en la articulación final, equivaldría a la muñeca. Su alcance es de 500 mm y su huella, es decir, el espacio que ocupa en el espacio de trabajo, es de 128mm.

2.1.2. Lenguaje de programación

Universal Robot apuesta por el lenguaje URScript y ofrece tres formas de programación con este lenguaje. Dos de estas formas se diferencian por la forma de programar y su complejidad, la tercera es una combinación de estas dos, dónde a partir de la programación intuitiva, permite llamar funciones programadas a través de scripts [8]:

- **Programación PolyScope**: programación intuitiva que se llevar a cabo a partir de la consola de programación del robot. Basado en bloques y un interfaz muy sencilla e intuitiva
- **Programación por Scripts**: permite crear programas más complejos y avanzados.
- **Programación por UR Caps:** una combinación de la programación intuitiva de PolyScope con la opción de añadir parte de programación directamente con código URScript lo cual permite desarrollar aplicaciones más complejas.

2.1.3. Entorno de programación y simulación

La propuesta del fabricante UR es poner a disposición una máquina virtual que simula con exactitud la consola de programación. Cuando se ejecuta la máquina virtual URSim (Universal Robot Simulator), permite escoger el modelo del robot que se quiere programar. Esta función es una herramienta de gran utilidad en el ámbito educativo y el ámbito profesional dado que permite crear programas sin necesidad de tener todo el conjunto de los dispositivos de entrada, simular su funcionamiento e instalarlo sin necesidad de parar la producción o la línea que contiene el robot.
Importante mencionar que, a diferencia del robot de verdad, no hace falta ponerlo en modo remoto, ya que, si no, no se podría ver como se mueve el robot. Simplemente se deja en local y el robot va haciendo caso a las órdenes. Lo que sí que hay que mirar es la dirección IP que tendrá este robot en el simulador, una vez encendido, para que se pueda realizar la comunicación por TCP/IP.

2.1.4. Comunicación

A partir del puerto de conexión Ethernet es posible establecer cuatro tipos de comunicaciones: Modbus TCP, Profinet y TCP/IP.

- **MODBUS TCP**: es uno de protocolo de comunicación diseñado específicamente por aplicaciones industriales destinado a la supervisión y control de equipos de automatización que permite una comunicación en el entorno de Internet utilizando la arquitectura cliente/servidor, el protocolo TCP/IP.
- **POFINET**: es un estándar industrial desarrollado por PROFIBUS International, para el intercambio de información a través de la red Ethernet mediante la lectura de datos y control de diferentes dispositivos.
- **TCP/ IP**: la comunicación por TCP/IP es un mecanismo de comunicación bidireccional entre dos dispositivos o dos procesos de un mismo programa basada en la filosofía cliente-servidor.

De las comunicaciones llamadas anteriormente se ha escogido trabajar con la comunicación TCP/IP porque es el único que nos ha permitido establecer una comunicación inalámbrica entre el robot y los microcontroladores.

2.1.5. Entradas y salidas

Los puertos de entrada/salida y los puertos de comunicaciones que disponen se pueden encontrar en la caja de control y en el extremo del brazo articulado en el conector Lumberg de 8 pines.
En la caja de control, todos los pines de conexión de entradas y salidas y conexiones de elementos de seguridad y accionamiento, trabajan con una tensión de 24 V. Los conectores amarillo situados en la primera fila de la parte izquierda permiten conectar botones de parada de emergencia que funcionan paralelamente con el botón de parada de emergencia que hay de la consola de programación. Los conectores situados a su derecha permiten conectar botones u otros sistemas. Por otro lado, los que están situados de forma horizontal, son puertos utilizados para conectar una cinta de transporte y poder sincronizar el movimiento de este con el robot y permitir un trabajo síncrono sin necesidad de parar la cinta. Las siguientes cuatro filas de conectores de color amarillo, permiten conectar dispositivos de seguridad que actúen como entradas de señales. Los siguientes conectores de color gris corresponden a pines de conexión de señales digitales. Por último, el conector de color verde permite conectar dispositivos analógicos.
En la parte inferior de la caja que contiene los dispositivos de control, se pueden encontrar (de izquierda a derecha): el conector por la consola de programación, una ranura por la tarjeta de memoria que contiene el sistema operativo, puerto de conexión Ethernet RJ-45, dos puertos de

conexión USB (USB2.0 y USB3.0) y un puerto de conexión mini DisplayPort.
La conexión Ethernet, se realiza mediante la conexión de un cable que va directamente al router que se ha instalado para poder realizar la comunicación inalámbrica por conexión wifi.

Ilustración 5. *Interior controlador robot UR3e y los puertos de conexión. Fuente: Universal Robots Academy.*

3. Dispositivo

En este apartado se va a hablar de las elecciones de los dispositivos que se han escogido para realizar este proyecto.

3.1. Robot

A diferencia de la gran mayoría de robots industriales, este robot se puede controlar a partir del protocolo de comunicación TCP/IP, además de tener API (Interfaz de Programación de Aplicaciones). El API es una herramienta que permite a los desarrolladores crear aplicaciones personalizadas para el robot.

Ilustración 6. *UR3e. Fuente: Universal Robots e-Series Manual de usuario*

3.2. Dispositivos de comunicación

En este punto en un primer momento se planteó usar un Arduino que tuviera ya un IMU incorporado en la propia placa y que tuviera la posibilidad de comunicación Wifi para poderse comunicar con el robot. Estos dispositivos son:

- **Arduino uno wifi rev2:** Esta placa tiene un acelerómetro de 3 ejes, un giroscopio de 3 ejes. Además, cuenta con un módulo NINA-W102 Wi-Fi y Bluetooth de u-Blox [24].
- **Arduino Nano 33 IoT:** Esta placa tiene un acelerómetro de 3 ejes, un giroscopio de 3 ejes y un módulo WiFi y Bluetooth integrados en el chip LSM6DS3 [9].
- **Arduino MKR WiFi 1010:** Esta placa no tiene acelerómetro, pero tiene un módulo Wifi y Bluetooth [10].
- **Arduino Nano RP2040 Connect:** Este modelo tiene un acelerómetro de 6 ejes, un giroscopio de 3 ejes, y tiene un módulo Wifi y Bluetooth integrados en el chip LSM6DSO [11].

Al final se eligió el Arduino Uno Wifi Rev2, porque este dispositivo es uno de los que se pudieron facilitar sin la necesidad de tener que comprarlo a priori, aparte de esto tiene la posibilidad de poder acoplar una shild encima. Como desenlace de varias pruebas básicas se llegó a la conclusión que ninguno de estos IMU mencionados anteriormente tiene un magnetómetro, es por eso que se ha descartado usar el IMU de estas placas. Pero sí usar su capacidad de conexión wifi.

Ilustración 7. *Arduino Uno Wifi Rev2. Fuente: Arduino*

3.3. Dispositivos de detección inercial

Como ya se ha mencionado anteriormente, el IMU que tienen incorporados los Arduinos con comunicación wifi integrada, no son suficiente para este proyecto. A continuación, se nombran 3 sensores IMU distintos a los anteriores ya mencionados:

- El **MPU-6050**: es una IMU de 6 grados de libertad, fabricado por Invensense, que combina un acelerómetro de 3 ejes y un giroscopio de 3 ejes. Es uno de los IMUs más famosos por su gran empleabilidad y, gracias a su calidad precio. Se puede conectar a través de I2C o SPI.
- El **LSM6DS3TR-C**:es un sensor IMU de 6 grados de libertad, fabricado por STMicroelectronics, que combina un acelerómetro de 3 ejes y un giroscopio de 3 ejes. Se puede conectar a través de I2C o SPI.
- El **BNO055**: es un IMU de 9 grados de libertad, fabricado por Bosch, que combina un acelerómetro de 3 ejes, un giroscopio de 3 ejes y un magnetómetro de 3 ejes. A diferencia del resto de IMU's mencionados anteriormente, este sensor también proporciona información sobre la orientación absoluta del dispositivo en el espacio gracias al magnetómetro.

Una vez se sabe qué microchip cumple con las necesidades que hay, se ha realizado una búsqueda para seleccionar un dispositivo que sea compatible con la placa Arduino Uno Wifi Rev2 [12] y tenga el microchip BNO055. Con esas características se ha encontrado la placa Arduino Shield 9 Axis Motion.

Ilustración 8. *Arduino Shield 9 Axis Motion. Fuente: Arduino*

4. Programa

4.1. Herramientas de programación

En este proyecto, para programar el microcontrolador y sus comunicaciones, se han usado dos Arduino uno wifi rev2. Cada uno de ellos contiene su propio Shield 9 axis motion, que es el que contiene el sensor IMU, el nombre del cual es BNO055.
Para la programación de este proyecto se ha optado por utilizar Visual Studio Code, un editor de código fuente gratuito y de código abierto, creado por Microsoft, el cual, compatible con múltiples lenguajes de programación y sistemas operativos, lo que lo hace muy versátil. Lo interesante de este programa, es que le puedes añadir extensiones, dependiendo del tipo de lenguaje o también ayudas para programar. En este caso, se ha usado para programar en Arduino, a través de PlatformIO.

Otra herramienta utilizada en el proceso de programación ha sido GitHub. Esta herramienta es una plataforma de alojamiento de código fuente y control de versiones basada en la web. En este proyecto en concreto, se ha usado

solamente para el almacenamiento de códigos generados para este, ya que, tiene dos cosas muy interesantes: la posibilidad de guardar la primera versión e ir subiendo mejoras al código máster y que se almacena el código con librerías y todos los archivos del código incluidos. Esta herramienta web también tiene la opción de sincronizarse con la aplicación Visual Studio Code, haciendo así más simple el proceso de guardar tu código en la nube.

Programas para simular

En la página de Universal Robot [13], nos ofrecen un simulador del robot, completamente gratis. Este simulador, llamado URSim, es un software de simulación que se utiliza para la programación y simulación fuera de línea de programas de robots. Cómo este programa solo es compatible con el sistema operativo Linux y, para este proyecto, se ha hecho uso de un sistema operativo Windows, Universal Robot, ha creado una máquina virtual para los usuarios de UR. Una vez que se tiene la máquina virtual con el programa de simulación, se necesita un reproductor virtual para la máquina virtual de UR. Existen dos opciones más utilizadas [13]:

- **VMWare Player** (gratis para uso no comercial, pequeña tarifa para uso comercial): Comentar que la versión de VMWare requiere una actualización a VM12.
- **VirtualBox** (programa gratuito): Esta opción no necesita de actualización, además de ser gratuito para todo tipo de aplicaciones. Por estos motivos ha sido el programa utilizado para la máquina virtual URsim.

4.2. Librerías

Las librerías principales en este proyecto son las siguientes:

- **Arduino NineAxesMotio**: con esta librería se va a leer todos los datos del chip BNO055. Utiliza la librería Wire.h, para poder hacer la comunicación por I2C. Además, tiene internamente la librería del propio chip de Bosch que sería la librería BNO055. A esta librería se le han tenido que hacer algunas modificaciones en el .h y en el .cpp. Al mirar en la librería, se pudo ver que no existía ninguna manera de que se lean los movimientos pitch, roll y heading en radianes en vez de en grados, por lo tanto, se modificó la librería par que así fuera.
- **WifiNINA**: Es una librería de Arduino que permite la conexión a redes inalámbricas utilizando el módulo NINA W101. Esta librería le permite utilizar las capacidades Wifi de Arduino Uno Wifi Rev2, Arduino Nano 33 IoT, Arduino MKR 1010 y Arduino MKR VIDOR 4000 WiFi. Además de poder instalar servidores, clientes y enviar/recibir paquetes UDP y TCP/IP a través de Wifi. La librería WifiNINA es la recomendada por Arduino para usar el Arduino Uno Wifi Rev2 [14].

4.3. Comunicaciones entre el dispositivo y el controlador del Robot

Uno de los objetivos es evitar comunicaciones físicas entre los dispositivos usados, evitando así dificultades motrices. Por lo tanto, para escoger la mejor comunicación se intentará evitar comunicación física siempre y cuando la trasmisión de datos sea rápida.

En primer lugar, se va a comenzar por el sensor que está insertado en la Shield 9 axis motion. Este sensor se tiene que conectar físicamente encima de Arduino uno wifi rev2, y su comunicación es por I2C [12]. Comentar que no había otra opción de comunicación, pero este tipo de comunicación física no provocará dificultades para la movilidad, por lo tanto, cumplimos con uno de nuestros objetivos.

La siguiente comunicación que tendría que haber entre dispositivos es la del Arduino Uno wifi rev2 con la del robot UR3e. Una vez sabemos los distintos tipos de comunicación que tienen cada uno. Podemos ver que la única comunicación factible entre estos dos dispositivos tendría que ser por TCP/IP. De esta manera se cumple el objetivo de evitar hacer comunicaciones físicas entre dispositivos. Para la comunicación por TCP/IP, se necesita crear una conexión como cliente con el servidor del robot, además de tener que usar también la dirección IP del robot.

Esta primera explicación de las comunicaciones entre estos dispositivos es un primer intento para conseguir el objetivo principal que es que el robot imite le brazo humano, pero realizándolo con un solo IMU. Al ser imposible realizarlo con un solo sensor, se hizo otro planteamiento en el cual entra en juego otro sensor más. Por ese motivo, ahora se tiene que plantear una nueva comunicación. Para la comunicación de este escenario, se ha preferido usar otro Arduino Uno Wifi Rev2 para poder usar un protocolo de comunicación inalámbrico. Hacer mención que uno de los dos Arduino estará comunicado por TCP con el robot, ya que, como se ha comentado anteriormente, es la única comunicación factible.

Después de hacer el estudio de varios tipos de comunicación, se llega a la conclusión que el método mediante comunicación Serial, finalmente ha sido efectivo para cumplir con el objetivo final de que el robot imite el brazo humano. Una vez averiguado esto, se procedió a crear un protocolo de comunicación por el puerto serial, entre los dos Arduino para que siguieran los pasos y se comunicasen los dos Arduino por orden.

Por último, es necesario explicar la configuración del router que se va a usar. Para mayor comodidad, se ha modificado el nombre y la contraseña de serie. Además, se ha configurado el router para que ponga la misma dirección IP al robot UR3e, y así no tener que estar cambiando la IP asignada en el código, para identificar el robot cada vez.

4.4. Obtención de datos del sensor

En este proyecto se ha utilizado el IMU BNO055, que está incorporado dentro del Arduino shield 9 axis motion, el cual incorpora tres sensores: el Magnetómetro, Acelerómetro y el Giroscopio, entre otros.

En un primer lugar, para usar estos tres sensores debe tener en cuenta que se pueden calibrar. Comentar que cada sensor tiene unos pasos específicos a seguir. Para ello, es necesario saber en qué nivel de calibración se encuentra el elemento. Siendo 0 nada y 3 totalmente calibrado.

4.5. Metodología

A continuación, se explicará la metodología seguida para la ejecución del movimiento del brazo robot. Un punto importante a tener en cuenta en este proceso es la implementación de un pulsador para controlar la transición

entre el proceso de calibración y la toma de datos del sensor. Por esa razón, se ha instalado un botón en la Shield para que, una vez el sensor esté calibrado, se tenga el tiempo suficiente para colocarlo en el brazo humano, y así poder empezar con toda la toma de datos en la posición correcta del sensor y, poder evitar problemas.

Primer método

Se intentó hacer la lectura de la posición del brazo humano con un solo IMU, utilizando principalmente los sensores del acelerómetro y el magnetómetro. Al observar problemas con las lecturas de datos y que por ello los cálculos generados no son totalmente lo realistas que se necesitan, se prosigue a realizar un segundo método.

Segundo método

En un primer momento, ya se optó por utilizar las lecturas de los ángulos de Euler que son el Roll, Pitch y Heading/Yaw, pero se tendrá que utilizar dos sensores en vez de uno. Estos ángulos de Euler son una forma común de describir la orientación tridimensional de un objeto o sistema en el espacio. A continuación, se explicará cada uno de los tres movimientos que hay:

- **Roll**: también llamado ángulo de balance o ángulo de inclinación, al medir los ángulos respecto del eje x del dispositivo. Esto quiere decir que es el ángulo en el que un objeto o sistema gira alrededor de su eje longitudinal, este es la línea que atraviesa el objeto de extremo a extremo.
- **Pitch**: también llamado ángulo de cabeceo o ángulo de elevación, al medir los ángulos respecto del eje y, del dispositivo, es decir, es el ángulo en el que un objeto o sistema gira alrededor de su eje lateral, que es la línea que atraviesa el objeto de lado a lado.
- **Heading o Yaw**: también llamado ángulo de guiñada, al medir los ángulos respecto del eje z, del dispositivo. Esto quiere decir que el ángulo en el que un objeto o sistema gira alrededor de su eje vertical, está en la línea que apunta hacia arriba desde el objeto. Esto hace que mida la dirección en la que el objeto está apuntando en el plano horizontal.

El primer contacto con estas tres lecturas de ángulos se hizo conectando físicamente a unos servomotores paso a paso, para poder ver bien si el movimiento que se generaba en el sensor era lo que se movían los motores paso a paso. Hay que tener en cuenta que el movimiento máximo de estos motores es de 180º como mucho.

En un primer momento se consiguió hacer el control de los motores aplicándole los ángulos leídos por el sensor. Pero a partir de cierto punto de giro se podía ver que el motor hacia movimientos incoherentes con el movimiento del sensor. Eso podía ser por el límite del motor y por el rango del sensor. Por ese motivo se, fue a investigar el rango de ángulos en el que se está dando cada uno de los tres movimientos.

Rango de ángulos del sensor:

Una vez ya se conocen las funciones de como determinar los grados de movimiento de los tres ejes y conseguir uno de los objetivos (detectar todos los movimientos del brazo humano), se tiene que investigar que rango de movimiento en ángulos son capaces de leer. Para ello, en un primer momento, se hizo una toma de datos girando en los tres ejes hacia el mismo sentido, para ir tomando los datos y poder determinar su rango en cada eje.

En primer lugar, se cargó el código que está en el *Anexo IV Código* en el microcontrolador, para controlar la lectura del BNO055, es decir, el sensor. Entonces, una vez cargado el código por el monitor serie, se pueden observar por pantalla los ángulos de los tres ejes. En un primer momento, se parte de un estado inicial con el sensor totalmente horizontal encima de la mesa. En este punto se hace la toma de los tres ángulos en cada uno de sus ejes en el estado inicial.

Una vez partido de este inicio, se va uno por uno girando y apuntando los diferentes valores dados, para poder hacernos una idea del rango de cada uno de los movimientos.

Una vez hecho esto y tener unos valores angulares aproximados, ya se puede deducir, viendo los valores leídos en cada caso, el rango que tiene cada movimiento. El rango de Pitch (eje X) va de 180º a -180º, el rango de Roll (eje Y) va de 90º a -90º y el rango de Heading (eje Z) de 0º a 360º.

Esta información, se ha podido corroborar posteriormente con el datasheet del sensor BNO055 que está en el según en el *Anexo III Esquemas y datasheets de los dispositivos*, el cual determina el rango de los tres movimientos.

Una vez ya se conocen los rangos angulares de los tres movimientos que se obtienen del sensor, hay que saber cuáles son los del robot para poder sincronizarlos correctamente.

Rangos de ángulos del Robot

Para empezar, hay que ser consciente de que el robot tiene 6 grados de libertad. Cada uno de ellos podría tener un rango distinto. Por ese motivo, en primer lugar, hay que revisar su ficha técnica que está en el *Anexo III Esquemas y datasheets de los dispositivos*.

Se puede observar que todos los motores, menos el de la Muñeca 3, tienen un límite de 2 vueltas completas y el de la muñeca 3 es infinito. Estos rangos van de 360º a -360º. El de la muñeca 3, almacena la cantidad de vueltas que lleva para un sentido o para el otro.

Determinar la posición inicial del robot

Una vez que se tienen los dos tipos de rangos que se tiene que sincronizar, hay que decidir que postura del robot se va a definir como postura inicial. Esta postura tendrá que ser la misma en el robot que en el brazo. Para llevar a cabo esta decisión hay que saber las limitaciones físicas del robot, de la similitud y de las diferencias de un brazo robot con respecto a un brazo humano. Ahora, fijándose en estas observaciones y con varias pruebas realizadas, incitó con el robot físicamente. Se ha acabado fijando como posición inicial el brazo humano completamente extendido en horizontal, determinado así el símil de esa postura con respecto al brazo robot. De esta manera, todos los movimientos del brazo humano son posibles de ser realizados por el brazo robot. En la siguiente imagen se puede observar dicha posición inicial.

Ilustración 9. *Posición inicial del UR3 e. Fuente: Elaboración propia*

Detección del brazo humano

En primer lugar, hay que ser consciente de la necesidad de dos chips BNO055, para poder detectar todas las posiciones del brazo humano. Porque, como ya se ha explicado, los movimientos Pitch, Roll y Heading detectan el ángulo de tres ejes de libertad, por lo tanto, quedan otros tres para llegar a los seis que son los que tiene un brazo.

Para la correcta lectura de los grados de libertad hay que tener muy en cuenta dónde están esos ejes de libertad y, colocarlos en el sitio correcto para que cada una de las lecturas sea una lectura útil.

El primer sitio donde se pueden generar tres movimientos diferentes es la muñeca, es decir, los tres posibles movimientos que se pueden hacer con esta parte del cuerpo se pueden realizar con respecto a tres ejes distintos unos de los otros. Para el resto del brazo, se pueden identificar tres articulaciones distintas: el giro del tronco, el hombro y el codo. Podemos definir cada uno de ellos con un movimiento distinto.

Con esta premisa, el primer IMU estará situado en la palma superior de la mano, detectando los tres últimos grados de libertad y, el otro, estará colocado en la parte superior del codo. Para hacerlo más fácil a continuación se puede la posición exacta.

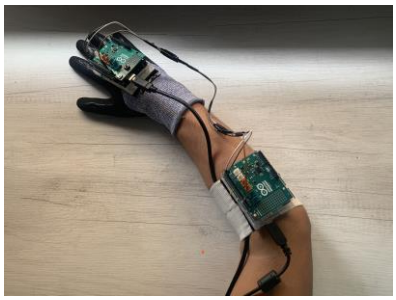

Ilustración 10. *Posición de los IMU en el brazo humano. Fuente: Elaboración propia*

Sincronismo con los dos brazos

En este paso se deber tener en cuenta toda la información recolectada en los apartados anteriores, los rangos de los ángulos de los dos dispositivos, tanto del sensor como del robot, y la posición inicial del robot, e implementarla toda junta para que el movimiento de ambos brazos sea igual.

Algo bastante importante es que el brazo robot no tiene los tres primeros motores con un movimiento con respecto a cada uno de los tres ejes (ejes x, y, z). Es decir, que los motores del codo y del hombro del robot se mueven los dos respecto del eje x. Eso significa que hay un movimiento que no podrá imitar el robot físicamente, el Roll.

Conociendo esta limitación, se ha elegido que el movimiento Pitch controle el motor del codo. La elección ha sido meramente por movilidad para poder realizar tareas con el robot.

El siguiente punto es, sabiendo los grados en que está el robot en esa posición inicial y su rango, se tendrá que adaptar los valores que lee el sensor con los que tiene el robot para que se muevan con la misma referencia angular.

Los valores angulares que tiene cada uno de sus motores en la consola del UR3e, son los siguiente [0, -π rad , 0, 0, +π/2 rad, 0]. La comparativa con los movimientos quedaría asi [Heading Hombro, será siempre -π rad, Pitch Codo, Pitch, Heading, Roll]. Como se ha comentado anteriormente, el de -π rad, se va a mantener estático. Ahora igual, pero con los rangos.

- **Rangos del robot** [-2π rad a 2π rad, -π rad, -2π rad a 2π rad, -2π rad a 2π rad, -2π rad a 2π rad, -∞ a ∞]
- **Rangos en el que los sensores enviarían sus datos** [0 a 2π rad, -π rad, π rad a -π rad, π rad a -π rad, 0 a 2π rad, -π rad a π rad].

De esta manera más visual se puede ver que todos menos los dos Heading, tienen los rangos de los sensores compatibles con los del robot, pero con menos recorrido. Para obtener el rango del Heading Hombro, se deben hacer cálculos para que en vez de ser de 0 rad a 2π rad sea π rad a -π rad, mientras que el otro Heading, debido a la posición inicial tendrá que marcar como si estuviera en π/2 rad.

En el código se ha generado una función par el Heading Hombro, para hacer los cálculos que pasa los datos que van de 0 rad a 2π rad, pase de π rad a -π rad.

Punto de referencia de los grados

Por último, e igual de importante, hay que saber el punto de referencia de cada uno de los movimientos.

Lo que se sabe es que el punto de referencia del Heading está referenciado a partir de donde está el norte. Lo encuentra con el magnetómetro. El Pich y el Roll, están referenciados a partir del acelerómetro que detecta la aceleración de la gravedad, haciendo que la horizontal sea su punto de referencia.

Ahora se tienen que realizar los cálculos para que se tenga el punto de referencia con respecto de la posición inicial, pero como la posición inicial del robot es en horizontal, solo habrá que cambiar el punto de referencia de los Heading. Donde se va a tomar los valores iniciales del sensor para poder cambiarlo por su punto de referencia en vez del norte. A continuación, en el Heading Muñeca 2 se le tendrá que sumarle los π/2 rad que tiene el robot en su posición inicial en el motor de la muñeca 2. Este proceso no es solo sumarle π/2 rad, sino que hay que cambiar el punto de referencia a π/2 rad.

En este punto, hay que tener en cuenta que el sensor de la muñeca puede detectar los ángulos del otro sensor. Para poder eliminar esas lecturas que no interesan que lea, porque se acumularán a su movimiento, se tendrán que restar.

El siguiente paso ha sido probar el código realizado, donde se ha detectado que el sentido de giro esta cambiado el del brazo con respecto el del sensor. Al identificar qué les ha pasado a todos los motores, hay que hacer los cálculos correspondientes para que los valores sigan el orden opuesto al que estaban haciendo hasta ahora. Hay que tener en cuenta los rangos y sobre todo el de la muñeca 2 que tiene π/2 rad en el estado inicial, ya que es el único que no se soluciona cambiando el signo.

Hay que mencionar que en un primer momento se tenían dudas de si sería necesario implementar un real-time, con una interrupción de tiempo. Pero al ver que la parte de ejecución del código no tardaba más de 31 milisegundos y que el puerto del servidor 30003 es de 1/500Hz= a 2 ms, en vez del resto de puertos que van a 1/10Hz = 0,1 s. Se observó que eligiendo el puerto 30003 ya no habría ningún problema debido al tiempo de ejecución del programa.

Flujograma

Gracias a todo el proceso anteriormente descrito y el flujograma obtenido, se puede desarrollar todos los pasos para realizar el código final. En el *Anexo V Flujogramas*, el Flujograma 3 es el flujograma del código final. Esto ha

ayudado a programar el código evitando errores de proceso, y posteriormente a la comprensión y entendimiento de las demás personas que tengan que trabajar con este código en un futuro. Además, también se puede encontrar los dos códigos finales de los dos Arduino están en el *Anexo IV Código*.

5. Presupuesto

A continuación, se pretende calcular el presupuesto aproximado total del proyecto, teniendo en cuenta todos los puntos anteriormente desarrollados.

Por un lado, se ha generado el presupuesto necesario para comprar todo el material necesario y este valor a dado **18.714,23€.**

Por último, también están contabilizadas las horas de ingeniería empleadas en el proyecto da un valor de **4.700€.**

Por lo tanto, el coste total de este proyecto, sumando todas las partes, tendría un valor de **23.414,23€.**

CONCLUSIONES

Una vez finalizado este proyecto se puede afirmar que cumple con las bases fijadas en la fase inicial. El objetivo principal de esta investigación, es conseguir que el brazo robot sea capaz de imitar el brazo humano, para poder llegar a realizar tareas a distancia sin la necesidad de estar en la misma sala que el robot, se ha podido desarrollar satisfactoriamente.

En este proyecto se ha podido observar la viabilidad y eficacia del control de los movimientos del brazo robot UR3e de Universal Robot, a través de la tecnología de los sensores inerciales y las distintas comunicaciones, haciendo posible la imitación del movimiento del brazo humano con el robot. Se ha obtenido una imitación muy realista del movimiento humano, demostrando así su viabilidad para aplicaciones futuras en entornos industriales y de investigación.

Aunque el proceso de calibración de los distintos sensores es algo engorroso, una vez calibrados y dentro de los parámetros, el sistema demuestra ser estable y de poder adaptarse a diferentes tareas.

Sin embargo, se han podido identificar ciertas limitaciones. En primer lugar, se han encontrado limitaciones angulares de los sensores con respecto el rango de movilidad del robot UR3e, ya que, existen diferencias físicas entre un brazo humano y el robot UR3e, aunque las dos tiene la misma capacidad motriz en el espacio. Además, la necesidad de tener el dispositivo del brazo humano conectado a un ordenador, dificulta el movimiento de este.

En conjunto, este proyecto aporta conocimientos para futuras investigaciones y desarrollos sobre el control de robots mediante imitación de movimiento, para así hacer una aportación al sector la automatización industrial y mejorar la interacción hombre-máquina en diversas aplicaciones.

AGRADECIMIENTOS

En primer lugar, quisiera dar las gracias a mi tutor de proyecto, Óscar de Sousa, y al equipo de STL (Serveis Tècnics de Laboratori), por ayudarme en todo lo que necesitara, enseñarme tantísimas cosas nuevas y hacer de este proyecto mucho más ameno y llevadero, disfrutándolo al máximo pese a su dificultad. También, quería agradecer a Antonio Camacho por introducirme en el mundo de los sensores inerciales y de la programación. Por último, agradecer a toda mi familia y amigos por su constante apoyo, ánimo y paciencia conmigo durante estos largos meses de duro trabajo.

REFERENCIAS

[1] Tipos de robots: clasificación, aplicacions y ejemplos. Disponible en: https://www.telefonica.com/es/sala-comunicacion/blog/tipos-de-robots-clasificacion-aplicaciones-y-ejemplos/

[2] Demostración y aplicación de la fórmula de Euler. Disponible en: ¿Qué dice la fórmula de Euler? | Demostración y aplicaciones (micalculadoracientifica.com)

[3] Euler angels, Wikipedia. Disponible en: Euler angles - Wikipedia

[4] Ángulos de Euler Yaw, Pitch, Roll. Disponible en: Ángulos de Euler Yaw, Pitch, Roll - programador clic (programmerclick.com)

[5] Universal Robots. Disponible en: https://www.universal-robots.com/es/

[6] E-Series OEM de Universal Robots. Disponible en: https://www.universal-robots.com/es/productos/oem-robots/

[7] Consola de programación e-series con dispositivo de validación de 3 posiciones. Disponible en: https://www.universal-robots.com/es/productos/e-series-3pe/

[8] Universal Robots, "The URScript Programming Language" 2013. Disponible en: https://www.zacobria.com/pdf/universal-robots-scriptmanual-en-v-1-8.pdf

[9] Nano 22 IoT, Arduino. Disponible en: **Nano 33 IoT | Arduino Documentation]**

[10] MKR Wifi 1010, Arduino. Disponible en: **MKR WiFi 1010 | Arduino Documentation**

[11] Nano RP2040 Connect, Arduino. Disponible en: **Nano RP2040 Connect | Arduino Documentation**

[12] 9 Axis Motion Shield, Arduino. Disponible en: 9 Axis Motion Shield | Arduino Documentation

[13] Offline simulator – e-series - URSim for non Linux 5.9.4, Universal Robots. Disponible en: https://www.universal-robots.com/download/software-e-series/simulator-non-linux/offline-simulator-e-series-ur-sim-for-non-linux-594/

[14] Host a Web Server on the Arduino UNO Wifi Rez 2. Disponible en: Host a Web Server on the Arduino UNO WiFi Rev2 | Arduino Documentation