UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DESARROLLO DE APLICACIONES MEDIANTE ROBOTS COLABORATIVOS BASADAS EN INTERFACES NATURALES HOMBRE-MÁQUINA

AUTORA: ELIZABET ZAKHARYAN

TUTORA: MARINA VALLÉS MIQUEL

COTUTOR: ÁNGEL VALERA FERNÁNDEZ

Curso Académico: 2017-18

AGRADECIMIENTOS

"A mi familia, por haber creído siempre en mi, haberme motivado y haberme dado todo su apoyo desde el principio de esta etapa tan enriquecedora de mi vida. Gracias por habérmelo dado todo."

"A Iris, por ser alguien incondicional desde el principio de este camino. Sin ella y sin su paciencia acabar esta etapa, llena de mil recuerdos bonitos gracias a ella, no habría sido posible."

"A Andrea, por seguir a mi lado y darme su apoyo incondicional siempre."

"A Marina y Ángel, por haberme ayudado y enseñado tantas cosas acerca de la Automática."

RESUMEN

En los últimos tiempos la robótica colaborativa está teniendo un gran auge puesto que se trata de robots mucho más económicos que los robots industriales tradicionales, son más fáciles de programar y facilitan el proceso de implantación puesto que no requieren costosos y complejos sistemas de seguridad. Esto provoca que la amortización de estos robots sea muy interesante para incluso pequeñas y medianas empresas.

En el presente Trabajo Fin de Grado se propone el desarrollo de aplicaciones con robots colaborativos, utilizándose para ello interfaces naturales hombre-máquina. Para ello se proponen desarrollar varias aplicaciones. En una de ellas se deberá establecer el control de los robots mediante reconocimiento de voz, de forma que se tendrá un sistema basado en tarjetas de control de bajo coste que deberá reconocer la voz del operario humano. Este sistema se conectará mediante comunicaciones inalámbricas a la unidad de control del robot colaborativo.

Una segunda interfaz natural que se deberá estudiar y trabajar es mediante la utilización de los sensores de fuerza que se disponen en las articulaciones del robot. A partir de la detección de la fuerza ejercida por el usuario, éste podrá parar y poner en marcha el robot en cualquier momento presionando ligeramente sobre cualquier parte de robot, lo que permitirá tener una interacción cómoda y segura con el robot.

Palabras clave: Robots colaborativos; interfaces hombre-máquina; aplicaciones de control por computador.

RESUM

En els últims temps la robótica col·laborativa està tenint un gran auge ja que es tracta de robots molt més econòmics que els robots industrials tradicionals, son més fàcils de programar i faciliten el procés d'implantació ja que no requerixen sistemes de seguretat costosos i complexos. Açò provoca que l'amortització d'estos robots siga molt interessant per a inclús xicotetes i mitjanes empreses.

En el present Treball Fi de Grau es proposa el desenvolupament d'aplicacions amb robots col·laboratius , utilizant-se per a això interfícies naturals home-màquina . Per això es proposen desenvolupar diverses aplicacions. En una d' elles s'haurà d'establir el control dels robots per mitjà de reconeixement de veu, de manera que es tindrà un sistema basat en targetes de control de baix cost que haurà de reconeixer la veu del operari humà. Este sistema es connectarà per mitjà de comunicacions sense fil a la unitat de control del robot col·laboratiu.

Una segona interfície natural que es s'haurà d'estudiar i treballar és per mitjà de la utilització dels sensor de força que es disposen en les articulacions del robot. A partir de la detecció de la força exercida per l'usuari, este podrá parar i posar en marxa el robot en qualsevol moment presionant lleugerament sobre qualsevol part del robot, la qual cosa permetrà tindre una interacció còmoda i segura amb el robot.

Paraules clau: Robots col·laboratius; interfícies home-màquina; aplicacions de control per computador.

ABSTRACT

In the last times the collaborative robotic is in a boom since robots are much cheaper than the traditional industrial robots, they are easier to program which facilitates its implementation process since they do not require expensive and complex security systems. Such features imply a great amortization which is very interesting even for Small and medium-sized enterprises (SMEs).

The present project proposes an applications development with collaborative robots with man-machine interfaces. To perform this, a several application development is proposed.

In one of them, a voice recognition will be established to control the robots, so there will be a system based on low-cost control cards that will recognize the voice of the human operators. This system will be connected through wireless communications to the control unit of the collaborative robot.

A second natural interface that should be studied and worked is by using the force sensors that are arranged in the joints of the robot. From the exerted force detection by the user, who can stop and start the robot at any time by lightly pressing on any part of the robot, which will enable a comfortable and safe interaction.

Key words: Collaborative robots; human-machine interfaces; computer control aplications.

ÍNDICE

DOCUMENTOS CONTENIDOS EN EL TFG

- Memoria
- Presupuesto

ÍNDICE DE LA MEMORIA

ÍNDICE DE PRESUPUESTOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

MEMORIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

1. INTRODUCCIÓN

En la actualidad, la mayoría de procesos industriales están o son susceptibles a ser automatizados. Mediante la automatización de cualquier proceso industrial, se consiguen grandes rendimientos con una gran economía de tiempo y costes. Por lo general, la automática no solamente está presente en los procesos industriales, si no en campos como la medicina, la aeronáutica, la automoción o la domótica, mediante la cual se puede automatizar una vivienda o edificio.

En la rama de la automatización industrial, una de las grandes y novedosas revoluciones dentro de ella, es la aparición de la robótica colaborativa y en el estudio de la cual se centrará este proyecto. La robótica colaborativa es el campo que abarca aquellos robots que son capaces de trabajar en un entorno interaccionando con los operarios facilitando y agilizando el trabajo de estos, sin ningún sistema de seguridad o protección de personas como las requeridas en los robots industriales. Los robots colaborativos o cobots entre otras características, son ligeros, sencillos de manejar y fácilmente programables. Cabe destacar que los cobots no sustituyen a los operarios, más bien colaboran con ellos para agilizar el proceso que está realizando el operario.

Por otro lado, el auge de la robótica colaborativa ha impulsado el crecimiento de la industria 4.0.

La industria 4.0 o también conocido como la “cuarta revolución industrial” es la digitalización de las industrias mediante la interacción de los procesos industriales con la robótica y la inteligencia artificial. La finalidad de dicha digitalización es optimizar los recursos, mejorar los procesos de producción y aumentar la rentabilidad de una industria.

Las ventajas que ofrece los cobots (robots colaborativos) en la industria 4.0 son muy amplias y variadas, y por ello, la robótica colaborativa es la que va a seguir impulsando el desarrollo y crecimiento de dicha industria en la actualidad.

Por un lado, los cobots (robots colaborativos) son fácilmente adaptables a diversos ámbitos dentro de una industria, con requerimientos mínimos de seguridad para interaccionar con humanos. Por otra parte, ofrecen un rápido ROI (retorno de inversión), lo que disminuye los costes de la industria al invertir en dichos robots. Además, la programación de estos cobots, al no requerir de personal con un nivel de especialización en automatismos elevada, hace que también se reduzcan costes de mano de obra.

Por tanto, el futuro de la industria a largo plazo, es la implantación masiva de cobots para realizar cualquier proceso industrial.

Finalmente, en este trabajo se profundizará en la programación de un brazo robot colaborativo de Universal Robots. La finalidad de dicha programación será implementar diferentes tipos de aplicaciones para procesos industriales controladas por voz o por fuerza.

2. OBJETIVOS Y MOTIVACIÓN

2.1. Objetivos

El objetivo principal del proyecto es controlar a través de dos diferentes formas de interfaces naturales hombre-máquina, el funcionamiento de un robot colaborativo.

La primera forma de controlar es a través del control por reconocimiento de voz, pudiéndose así controlar diferentes procesos programados en un brazo robot colaborativo.

La segunda forma de control se logra a través del control por fuerza, que consiste principalmente en aplicar una fuerza con la mano al robot (apretando ligeramente en cualquier parte del robot), para que comience a funcionar o se detenga en un determinado punto. En este método de controlar el funcionamiento se han empleado algunos de los procesos que se han programado también para ser controlados por voz.

Por otro lado, otro de los objetivos del proyecto es estudiar más profundamente el campo de la robótica colaborativa y las mejoras que se pueden implementar en ella. Así, con el objetivo principal del proyecto, se consigue potenciar las capacidades que nos ofrece la robótica colaborativa y mejorar la interacción que dispone con el ser humano.

Así pues, se puede conseguir una interacción con el robot más sencilla, cotidiana y directa para el ser humano, ya que para el ser humano la interacción se basa en acciones naturales y frecuentes, en este caso el habla y el tacto, que realiza en su día a día.

2.2. Motivación

La motivación de este proyecto es aplicar diversos conocimientos adquiridos durante el grado, como es la programación y la robótica para lograr llevar a cabo el desarrollo de las formas de control anteriormente mencionadas.

Además, otra de las motivaciones principales, es poder aplicar las formas de control desarrolladas en el proyecto a procesos reales que se llevan a cabo en la industria. Esto es posible gracias a que la autora de este proyecto está realizando prácticas en la empresa ISTOBAL S.A. donde mediante la robótica colaborativa se pretende automatizar procesos repetitivos que se llevan a cabo diariamente en la fábrica, para lograr un aumento de precisión y una reducción de tiempo y costes en dichos procesos. Por el momento se está automatizando el proceso de mecanizado de cajas, y está en fase de estudio automatizar el proceso de lijado de carenados de maquinaria para lavado de automoción, teniendo intención de automatizar más procesos que se detecten como repetitivos.

La idea final es aplicar los sistemas de control programados a estos procesos, de manera que se facilitará la interacción del operario con el brazo robot, bien sea dando órdenes a través de la voz o a través del control por fuerza. Por el momento, está en desarrollo aplicar el control por voz al mecanizado de cajas, así el operario simplemente con la voz ordenará al robot el tipo de caja a mecanizar y que el robot comience a funcionar o pare su funcionamiento.

3. ROBÓTICA

En la actualidad, la robótica es cada día más imprescindible para el desarrollo de la sociedad. Por ello, cada día es más notable la presencia de la robótica en las industrias, o a nivel doméstico en aparatos de uso cotidiano o incluso su presencia en el ámbito de la sanidad. Además, la robótica es un campo en constante desarrollo desde su creación, ya que, cada vez aparecen tecnologías más novedosas e innovadoras, que hacen posible que la industria actual avance y sea cada vez más eficiente.

Así pues, a continuación se detallarán las bases de la robótica y la forma en la que se pueden clasificar los tipos de robots que existen.

3.1. Historia de la Robótica

La aparición de la robótica industrial después de la Segunda Guerra Mundial con el desarrollo de manipuladores mecánicos para elementos radiactivos el cual se basaba en el sistema "maestro-esclavo". Dicho mecanismo conocido como *tele manipulador* se desarrolló en el 1948 por R.C. Goertz del Argonne National Laboratory.

Posteriormente, se desarrolló gracias a Ralph Mosher de General Electric, un dispositivo que consistía de dos brazos tele operados controlados mediante un dispositivo maestro.

Sin embargo, todos estos dispositivos funcionaban mediante control mecánico y surgió la idea, a raíz de la aparición del primer computador electrónico en 1946 o del desarrollo de dispositivos mecánicos con servomecanismos de posición y velocidad en 1948, de evolucionar de dicho tipo de control a un control mediante computadores.

Finalmente, tras unos años de avances, en 1959 tuvo lugar la aparición del primer robot industrial gracias a Unimation Inc. , empresa fundada por Engelberger y pionera en robótica. Este robot, utilizaba un computador junto a un manipulador con el cual formaban una máquina que se programaba para realizar varias tareas de forma automatizada.

En la década de los 60, concretamente en 1962, H.A. Ernst desarrolla una mano mecánica, llamada MH-1, con sensores táctiles controlados por computador. También en ese mismo año Tomovic y Boni desarrollaron una mano con sensor de presión para la detección de un objeto. Finalmente, en 1963, sale al mercado el robot comercial VERSATRAN diseñado por la American Machine and Foundry Company (AMF) y los brazos manipuladores Roehampton y Edinburgh.

En 1968, McCharthy en el Stanford Artificial Intelligence Laboratoy del Stanford Research Institute publica el desarrollo de un sistema con manos, oídos y ojos que podía reconocer mensajes hablados, detectar piezas distribuidas en una plataforma y manipularlas siguiendo unas instrucciones. Además, este mismo año se inicia la investigación y difusión de la robótica industrial en Japón.

Por un lado, en la década de los 70 se produce un apogeo en la investigación, implantación y desarrollo de la robótica. Así pues, en Europa las primeras aplicaciones industriales tuvieron lugar entre 1970 y 1971, en cadenas de fabricación de automóviles. Además, en 1973, ASEA (actual ABB), construyó IRB 6 el primer robot con accionamiento eléctrico.

Por otro lado, entre 1973 y 1974 Cincinnati Milacron diseña el T^3 (The Tomorrow Tool), un robot industrial controlado por computador que podía levantar aproximadamente 50 Kg y seguir objetos móviles en una línea de montaje. Además, en el 1976 los robots espaciales de la NASA, Viking 1 y Viking 2 fueron utilizados para tomar imágenes en Marte.

A mediados de los 70, General Motors financió un programa en el que el investigador Víctor Scheinman del Instituto Tecnológico de Massachusetts inventó un brazo mecánico denominado "manipulador universal programable para ensamblaje", mejor conocido como PUMA, que servía para tareas de producción. Este robot marcó el inicio de la era de la robótica industrial.

Ya en los 1980, se crea la Federación Internacional de Robótica (IFR) en Europa. Finalmente, en 1982 Makino de la Universidad de Yamamashi (Japón) crea el concepto de robot SCARA (Selective Compilance Assembly Robot Arm) que es un brazo robótico que posee 3 o 4 grados de libertad para ser utilizado en el ensamblaje de piezas.

En resumen, la aparición de la robótica se remonta a la década de los 50, pero no es hasta finales de los 70 o principios de los 80 cuando experimenta un auge, donde se comienza a comercializar los robots y con ello se logra automatizar las industrias, aumentando la producción y el rendimiento y disminuyendo costes y tiempo.

Por tanto, se puede afirmar con total certeza que la robótica industrial es una de las mayores innovaciones del siglo XX.

3.2. Clasificación de los robots

Se puede hacer una clasificación de los robots en función de la evolución que han tenido a lo largo del tiempo. Esta clasificación se puede hacer distinguiendo tres generaciones que se detallarán más detenidamente a continuación.

A lo largo del tiempo, los robots han tenido y siguen teniendo una evolución continua, con el objetivo de conseguir una mejora en las características y especificaciones técnicas del robot, para que tenga un rendimiento mayor en cualquier ámbito que tenga que desenvolver su trabajo.

En primer lugar, la primera generación de robots se desarrolló alrededor de los años 50, siendo estos robots manipuladores, pero sin capacidad de ser programados por un operario humano, en cambio, si podían ser dirigidos para realizar ciertas tareas, pero no conocían el entorno en el cual desarrollaban su función.

Tras esto, alrededor de los años 80 se pasa a la segunda generación (robots de aprendizaje), mediante la implantación de sistemas sensoriales en el robot manipulador, como sistemas de distancia, posición o visión, y la capacidad de ser programado anteriormente por un operario.

Por otro lado, la evolución continua hasta una tercera generación, como el robot de la Figura 1, en la cual los robots ya son capaces de desenvolverse en ámbitos de trabajo complejos. Actualmente, esta generación no está completa todavía porque se siguen creando nuevos prototipos y modelos.

Por último, respecto a la cuarta generación, son conocidos como robots inteligentes y a parte de recibir información del entorno para realizar la tarea, también son capaces de enviar información sobre el estado del proceso, es decir, intercambian información con la computadora de control.

Figura 1. Brazo robot industrial. [Fuente: http://step-automation.es/2-1-4-industrial-robot/226197/]

3.3. Tipos de robots

Actualmente, existen muchos tipos de robots, con diferentes características y funcionalidades, a continuación se detallarán los tipos de robots más conocidos y empleados.

Figura 2. Brazo industrial de manipulación de ABB. [Fuente: https://es.kisspng.com/png-industrial-robot-robotics-abb-group-robot-welding-4354448/preview.html]

Robots industriales de manipulación: Son robots con una base fija anclada a una plataforma de trabajo, por tanto, son robots articulados que desplazan un útil de trabajo por el espacio.

Además, pueden ser multifuncionales y reprogramables, y poseen un control absoluto del entorno en el que han de desenvolver su trabajo, sobretodo en ámbitos de producción realizando tareas que requieren mucho esfuerzo físico para el operario. En este tipo de robots, se encuentra el brazo robótico con el que se va a trabajar en este proyecto.

Figura 3. Robot de limpieza de hogar. [Fuente: https://tiendas.mediamarkt.es/p/robot-aspirador-irobot-roomba-695-wifi-1365921]

Robots de servicio: Son dispositivos controlados por ordenador, suelen ser móviles y sustituyen al hombre en procesos cotidianos. Son de uso individual, por ejemplo, los robots empleados en realizar tareas de limpieza como el de la Figura 3, que, además, son capaces de adaptarse a cambios en el torno en el que han de desenvolverse.

Figura 4. Robot quirúrgico. [Fuente: https://www.theexpertinstitute.com/the-da-vinci-robot-expert-witness-a-litigation-guide/]

Robots médicos: Actualmente, la robótica cada vez está más presente en el ámbito sanitario, facilitando muchas tareas al personal sanitario. Para diversas operaciones médicas, se emplean dispositivos láser con una precisión casi total o en el ámbito de la sanidad dental también se disponen de diversos tipos de aparatos, como el de la Figura 4, que facilitan y aumentan la precisión de las operaciones realizadas.

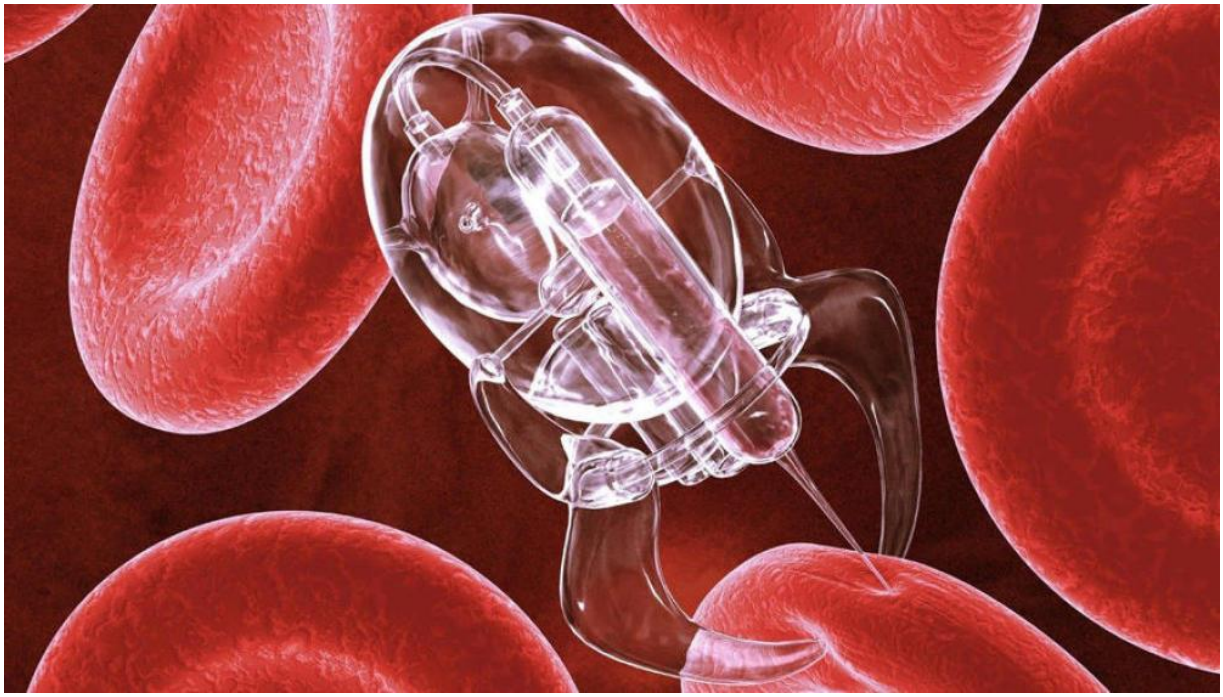

Figura 5. Nano robot. [Fuente: http://www.lasexta.com/noticias/ciencia-tecnologia/nanorobots-buscar-destruir-tumores-asi-tecnologia-que-salvara-vidas-futuro_201802125a81c2b30cf216bbfc6c2606.html]

Nano robots: Los nano robots todavía están en una fase muy prematura, sin embargo, son la tecnología y robótica del futuro, y ya se ha logrado diseñar nano robots, véase la Figura 5, que se insertan dentro del organismo humano y son capaces de buscar y destruir tumores.

3.4. Industria 4.0 en la actualidad

Figura 6. Esquema Industria 4.0. [Fuente: http://www.talleresmorte.com/morte-explores-the-possibilities-of-industry-4-0/?lang=en]

El concepto de industria 4.0 surge de la necesidad de digitalizar la industria, y todos los procesos que la componen, y conseguir mejoras sin precedentes. Esta digitalización, haciendo posible que la industria se transforme al concepto de industria inteligente, informatizando y automatizando todos los sistemas de producción y fabricación interconectados a través del internet de las cosas(IoT).

Así, mediante la digitalización, gracias a los avances en la tecnología electrónica, informática y automática, se consigue dar una flexibilidad y una independencia a la industria, que se traduce en resultados más eficientes que los obtenidos en la industria que existe hoy en día.

Como se puede observar en la Figura 6, la industria 4.0 está compuesta de muchos ámbitos, y todos ellos a través de una sinergia conjunta, logran dotar de inteligencia a la industria y sus procesos.

Así pues, unas de las bases de la industria 4.0, sin la cual la digitalización sería imposible, es la robótica en general. Sin embargo, este proyecto se va a enfocar en la robótica colaborativa, un campo que permite una clara interacción hombre-máquina de manera sencilla y óptima.

4. ASPECTOS TÉCNICOS DE UN ROBOT

Sin embargo, antes de profundizar en el concepto de robótica colaborativa y presentar el robot colaborativo protagonista de este proyecto, hay que definir los aspectos técnicos más importantes de un robot, que son cruciales ya que influyen en la capacidad de trabajo del robot, determinando así, su eficiencia a la hora de realizar tareas dentro de una industria.

4.1. Área de trabajo

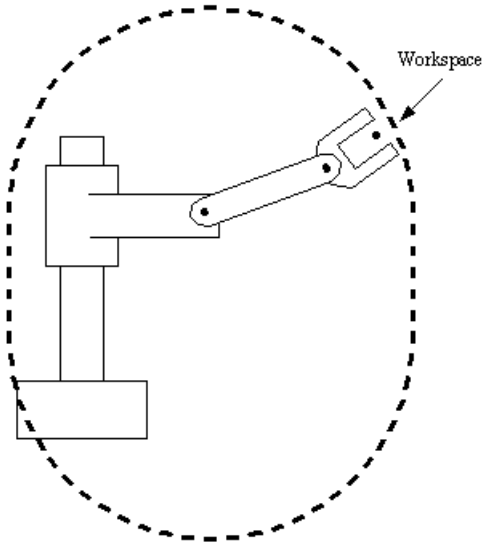

Figura 7. Área de trabajo de un robot. [Fuente: http://www-assig.fib.upc.es/~rob/protegit/treballs/Q2_03-04/general/carmorf.htm]

El área de trabajo de un robot es el volumen espacial que el extremo del robot puede alcanzar. Para los robots cartesianos, el espacio de trabajo es similar a un cuadrado, pero en el caso de los robots más modernos de la actualidad, pueden tener diversas formas geométricas ya sea elipses o formas esféricas como en la Figura 7.

Sin embargo, en ciertas zonas de ese volumen en el cual el alcance del robot está limitado en la orientación de la muñeca, ya que existen diseños mecánicos donde hay ejes de rotación que no pueden girar 360º.

4.2. Grados de libertad

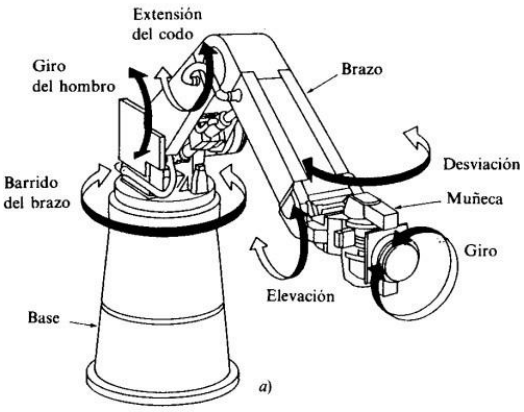

Figura 8. Grados de libertad de un robot. [Fuente: https://itziasite.wordpress.com/grados-de-libertad/]

El factor de los grados de libertad es esencial para saber la capacidad de orientación que puede tener el robot de la herramienta que lleve conectada en el extremo, ya sea una pinza o cualquier otro tipo de herramienta.

Los grados de libertad suelen coincidir con el número de articulaciones que posee el robot, ya que como se observa en la Figura 8, una articulación representa un grado de libertad. Cabe destacar, que existen robots con grados de libertad ampliables, en 1 grado o 2 a lo sumo, de modo opcional.

4.3. Capacidad de carga

La capacidad de carga es uno de los factores más importantes a la hora de elegir un robot determinado. Se puede definir como el peso máximo que puede transportar el robot, garantizando el correcto funcionamiento del brazo robot, mediante su pinza cuando realiza tareas.

Este factor depende del tamaño y del peso del robot, de su propia configuración y del sistema de accionamiento que se utiliza. Además, este factor disminuye cuanto más lejos esté de la base, ya que lejos de la base se crean momentos en el extremo del robot.

4.4. Precisión

La precisión se podría definir como la distancia entre el valor medio de los puntos que recorre el robot para alcanzar un punto programado y el punto programado.

Una mayor o menor precisión depende de factores como el modelo de control cinemático y dinámico, o deformaciones dinámicas y térmicas que puede sufrir el brazo robot. Además, cuanto más alejado esté el brazo de la base, menor será su precisión ya que cuando se extiende el brazo, las imprecisiones mecánicas aumentan.

4.5. Repetibilidad

La repetibilidad es el radio de esfera que contiene los puntos por los que pasa el robot al recibir la orden de ir al mismo punto programado. Un error en dicho parámetro es debido a problemas en el sistema de transmisión, como, por ejemplo, rozamientos o histéresis.

5. ROBÓTICA COLABORATIVA

Finalmente, y tras una breve introducción al mundo de la robótica y a su impacto en la sociedad, hay que enfocarse en el campo de la robótica colaborativa. Así pues, gracias a los avances en la tecnología, en especial, de la robótica industrial, es posible hacer realidad el innovador concepto de robótica colaborativa.

La robótica colaborativa aparece de la necesidad del ser humano de interaccionar con robots en el sector industrial, ya que la sociedad y sus exigencias aumentan cada vez más, siendo necesario optimizar los recursos disponibles, disminuyendo tiempos y costes de producción.

Sin embargo esta necesidad no era posible suplirla ya que los brazos robóticos disponibles en la industria no pueden interaccionar con un ser humano cuando están en funcionamiento porque supone un gran peligro para la persona interponerse en la zona de trabajo de dichos robots (suelen estar dentro de un espacio de trabajo inaccesible). Además, dichos robots son costosos, de gran tamaño y muy pesados, y disminuir todos estos factores era una de las principales motivaciones que perseguía el concepto de robótica colaborativa. Pero el factor más importante es romper con el peligro que supone la interacción de un ser humano con los robots industriales existentes hasta el momento.

Actualmente las empresas más pioneras en la industria robótica, como por ejemplo ABB Robotics o KUKA, que a continuación se verán con más detalle, también han desarrollado sus prototipos de brazos robóticos colaborativos, con el objetivo de mejorar la industria y todos los procesos que abarca. Esta mejora se puede traducir a un coste más reducido que con los brazos robóticos que ya disponían antes de la aparición de la robótica colaborativa.

5.1. ABB Robotics

Figura 9.Logo de ABB. [Fuente: https://en.wikipedia.org/wiki/ABB_Group]

ABB surge de la fusión de las compañías ASEA y BBC en el año 1988, y es una corporación líder en fabricación de robots industriales y sistemas robóticos a nivel mundial.

ABB Robotics decidió innovar presentando su prototipo de robot colaborativo en el año 2015, abriendo así un nuevo sinfín de posibilidades y mejoras en la industria actual.

5.1.1. YuMi®

Figura 10. Robot YuMi®. [Fuente: https://new.abb.com/products/robotics/es/robots-industriales/yumi]

Este robot colaborativo denominado **YuMi® (You and Me)** es como la recreación de un torso humano con dos brazos que poseen manos flexibles y, además, incluye un sistema de visión patentado y tecnología de la última generación, lo que hace posible disponer de un robot que según ABB Robotics *"cambiará nuestro concepto de la automatización del ensamblaje. YuMi® es como trabajar "con un compañero", con posibilidades ilimitadas."*

[Fuente: https://new.abb.com/products/robotics/es/robots-industriales/yumi]

Por otro lado, su programación es muy sencilla ya que YuMi dispone de una programación guiada que no requiere líneas de código, solamente se ha de colocar los brazos en diferentes posiciones y abrir y cerrar las pinzas para disponer de un programa en cuestión de poco tiempo.

Además, dispone también de detección de control de fuerza, haciendo que el robot pueda funcionar aplicándole un determinado valor de fuerza.

Este robot, está diseñado especialmente para procesos en los que se necesiten ensamblar piezas pequeñas, como por ejemplo, en el sector electrónico durante el montaje de placas electrónicas en las que junto al operario, es capaz de trabajar en un área de trabajo pequeño y recreando prácticamente el comportamiento humano, por lo que colabora "mano a mano" con el operario, sin ningún peligro ya que dispone de una "seguridad intrínseca" , disminuyendo tiempos de producción y aumentando el rendimiento de la tarea.

5.2. KUKA

Figura 11.Logo de KUKA. [Fuente: https://logos-download.com/12959-kuka-logo-download.html]

KUKA es una empresa pionera a nivel mundial en robótica e instalaciones industriales cuya historia se remonta al año 1898. Ellos mismos, en su página oficial, definen la empresa de modo que *"Hoy en día somos una de las empresas líderes en robótica y técnica de instalaciones y sistemas." [Fuente: https://www.kuka.com/es-es/acerca-de-kuka/historia]*

KUKA ofrece una gran variedad de robots industriales de última tecnología, e igual que muchas empresas pioneras en el sector de la robótica industrial, también ha desarrollado su propio prototipo de robot colaborativo denominado LBR iiwa.

5.2.1. LBR iiwa

Figura 12. Robot LBR iiwa. [Fuente: http://www.interempresas.net/Robotica/Articulos/156639-Kuka-vuelve-a-la-BIEMH-como-partner-experto-en-robotica-colaborativa.html]

Robot de estructura liviana, cuya abreviatura es LBR y, por otro lado, iiwa es la abreviatura de *"intelligent industrial work assistant".*

El LBR iiwa sigue el mismo principio en el que están basados los robots colaborativos, trabajar en tareas complejas cooperando con humanos sin ningún peligro o riesgo para estos.

Por otro lado, el LBR iiwa es adecuado para procesos tales como paletizado, embalaje, carga, operaciones de montaje o manipulación de otras máquinas entre otros.

5.3. Universal Robots

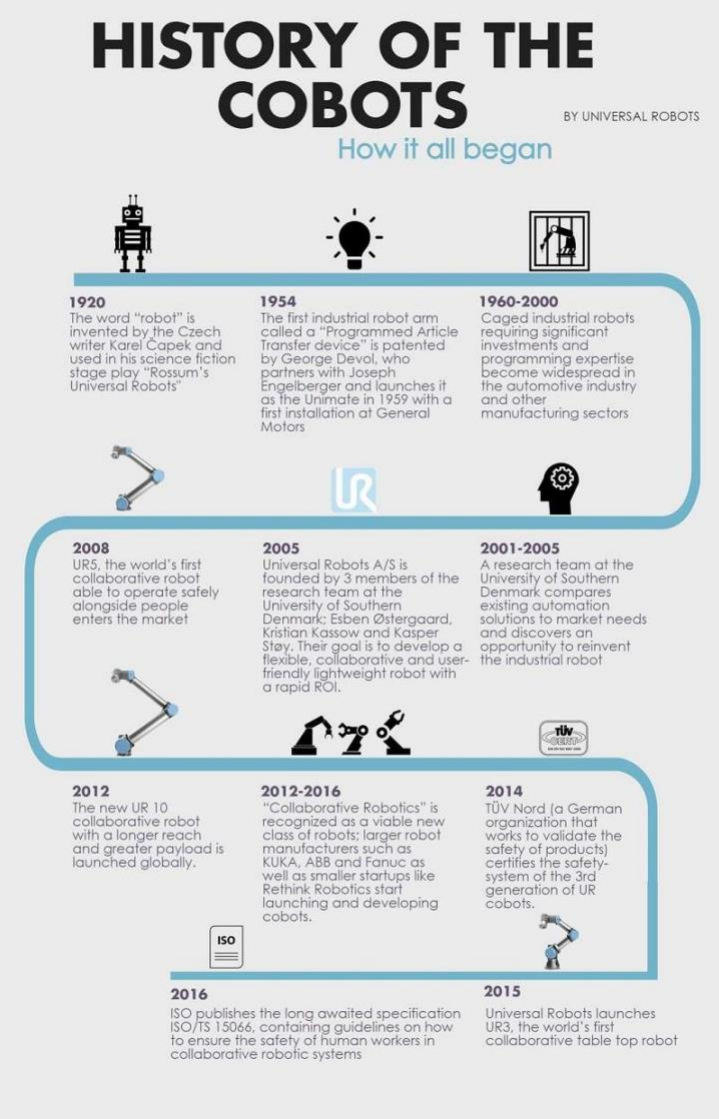

Figura 13. Cronología de Universal Robots. [Fuente: https://www.universal-robots.com/about-universal-robots/news-centre/the-history-behind-collaborative-robots-cobots/]

Finalmente, se centra la atención en la empresa del robot con el que se va a trabajar en este proyecto. Además, se puede observar en la Figura 13, la cronología y la historia de los cobots de Universal Robots.

Motivados por un afán de crear un robot innovador y que se adaptara a la industria actual, un grupo de investigadores de la Universidad del Sur de Dinamarca se propusieron reinventar la robótica y finalmente tres de ellos, fundaron Universal Robots en 2005.

Universal Robots es la primera empresa, fundada por Esben Østergaard, Kasper Støy y Kristian Kassow, en fabricar un brazo robot colaborativo y ponerlo a la ventana. En concreto, el primer modelo diseñado fue el UR5 en el año 2008, siendo sus sucesores UR10 en 2012 y finalmente UR3 en 2015.

Cabe destacar de la gama de robots de Universal Robots, que todos los modelos que disponen, poseen un diseño sencillo, son fácilmente programables ya que no requieren de conocimientos previos en programación, ni expertos en programación y su retorno de inversión (ROI) es notablemente rápido.

Además, esta gama de robots puede integrarse en cualquier tipo de industria, industrias de automoción, para fabricación de productos sanitarios, para industrias alimenticias o para industrias electrónicas. También es posible su acople sobre una plataforma de trabajo, optimizando las operaciones para las que está programado y adaptándose a cualquier ámbito de trabajo.

Universal Robots posee dos gamas de robots, los UR(3,5 y 10) y los URe-Series(e3,e5 y e10). Las dos gamas poseen características similares (la diferencia más notable es la mejora de la resolución en los e-Series). Sin embargo, los URe-Series, fusionan según Universal Robots , *"productividad, adaptabilidad y fiabilidad"* , y además Universal Robots los describe del siguiente modo:

"La gama e-Series es polivalente, fácil de programar y se puede integrar sin dificultad ninguna en todos los entornos de producción, sea cual sea el tipo de producto o planta de fabricación." [Fuente: https://www.universal-robots.com/es/e-series/]

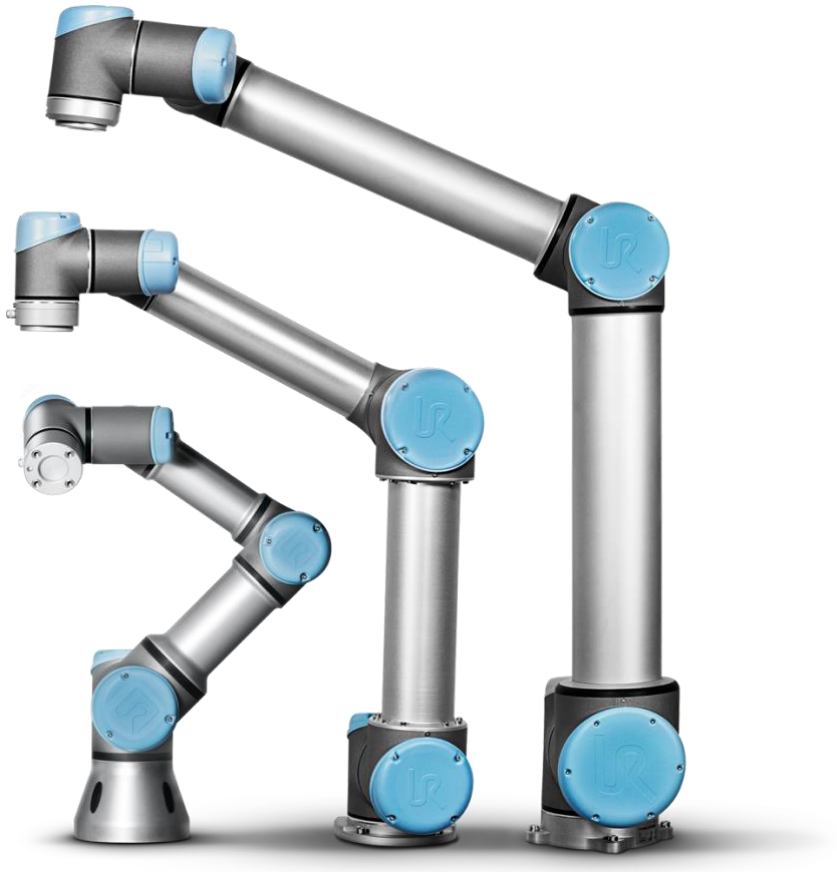

Figura 14. Gama Universal Robots: UR3, UR5 y UR10. [Fuente: https://cobotsguide.com/2016/06/universal-robots/]

Así pues, a continuación se explicarán más detalladamente las características y las posibilidades que nos ofrece cada tipo de robot de Universal Robots mostrados en la Figura 14.

5.3.1. UR3

La gama UR3 son los robots más pequeños de toda la gama de Universal Robots y actualmente son los robots más ligeros y flexibles de la industria. Por ello, son muy apropiados para procesos de montaje, pick&place, pulido, encolado y atornillado, ya que son rápidos y poseen una elevada precisión.

5.3.1.3. Características

5.3.1.3.1. UR3

·**Peso:** 11kg

·**Huella:** 128mm

·**Capacidad de carga:** 3kg

·**Alcance:** 500mm

·**Velocidad:** articulaciones de la muñeca: 360 grados/s

Otras articulaciones: 180 grados/s

Herramientas: Típico 1m/s

·**Repetibilidad:** +/- 0.1mm

·**Grados de libertad:** 6 articulaciones giratorias

5.3.1.3.2. UR3e

Mantiene todas las características igual, excepto el peso que aumenta a 11.2kg y la repetibilidad que disminuye de 0.1mm a 0.03mm.

5.3.2. UR5

Por otro lado, la gama UR5 es muy empleada para la automatización de procesos colaborativos de pesos pequeños como serían las pruebas de producto o el pick&place.

5.3.3. UR10

Finalmente, el robot UR10 es el brazo robot de mayor tamaño y peso de la gama UR. Este robot es el más empleado para procesos que requieran un mayor peso que los anteriores, como bien sería, el montaje de piezas, el empaquetado y el pick&place de piezas de mayor peso y tamaño.

Cabe destacar, que, gracias a su amplio alcance, se ahorra tiempo y, por tanto, costes en la producción, ya que se acortan las distancias entre la línea de producción y el robot.

6. MÉTODOS DE PROGRAMACIÓN DE LA GAMA UNIVERSAL ROBOTS

Actualmente, los robots de Universal Robots son programables a través de Polyscope o bien existe la posibilidad de la programación mediante scripts. No obstante, se profundizará más en la programación a través de Polyscope puesto que las aplicaciones desarrolladas en este proyecto son a través de dicho método, pero la programación mediante scripts también es una opción viable.

6.1. PolyScope

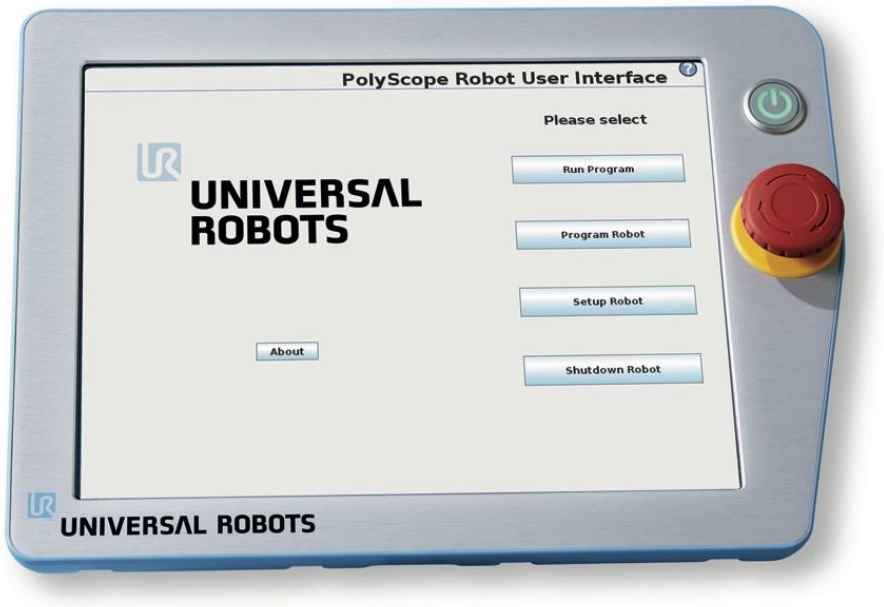

Figura 15. Interfaz gráfica de Universal Robots. [Fuente: http://www.nonead.com/en/download_content/610.html]

PolyScope es un software desarrollado por Universal Robots, constituyendo el interfaz gráfico con el usuario que ha de manipular el robot. Este programa está implementado en una Tablet de 12 pulgadas igual al de la Figura 15, mediante la cual se realiza toda la programación del robot.

6.1.1. Interfaz del usuario

Figura 16.Pantallazo Interfaz de usuario. [Fuente: https://www.universal-robots.com/media/207448/ur3_user_manual_es_global.pdf]

6.1.2. Pantalla de inicialización del robot

Figura 17. Pantallazo de inicialización del robot. [Fuente: https://www.universal-robots.com/media/207448/ur3_user_manual_es_global.pdf]

Mediante esta pantalla, se consigue la inicialización y el arranque del robot, simplemente apretando el botón de iniciar, y a partir de ahí, se liberan los frenos y finalmente, se enciende el robot.

6.1.3. Inicio

Figura 18.Pantallazo programa nuevo. [Fuente: https://www.universal-robots.com/media/207448/ur3_user_manual_es_global.pdf]

La interfaz de inicio es muy sencilla, ya que se puede optar por cargar un programa guardado en un dispositivo externo, como puede ser una memoria USB.

O bien, existen plantillas para crear un programa nuevo o una plantilla de un programa típico como sería cargar y descargar cualquier objeto.

6.1.4. Programa del robot

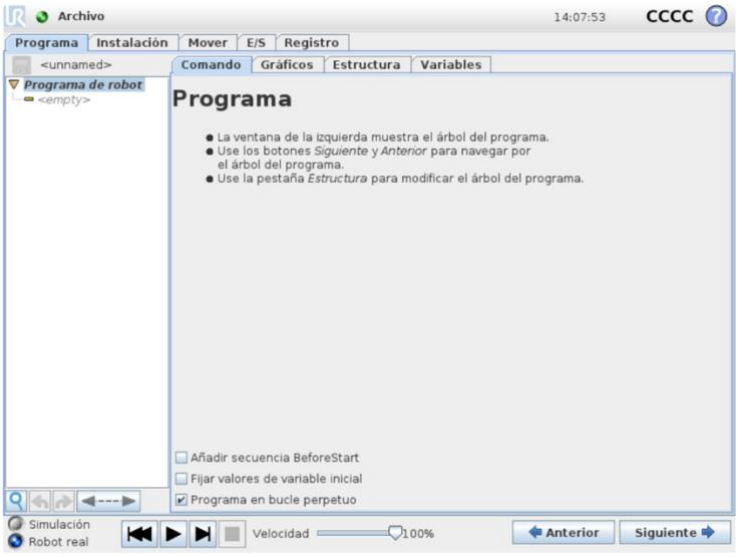

Figura 19.Pantallazo del programa del robot. [Fuente: https://www.universal-robots.com/media/207448/ur3_user_manual_es_global.pdf]

Como se puede observar en la Figura 19, el programa del robot estará implementado en el espacio en blanco del lado izquierdo, viéndose como una especie de diagrama de árbol, muy sencillo de comprender.

Además, en esta pantalla, se disponen de cuatro pestañas : comando, gráficos, estructura y variables.

En la pestaña gráfico, se puede ver el gráfico del robot.

En la pestaña estructura se tienen todos los comandos posibles para poder programar.

Finalmente, en la pestaña variables, aparecen todas las variables que posee nuestro programa.

6.1.5. Comandos de programación

6.1.5.1. Nivel básico

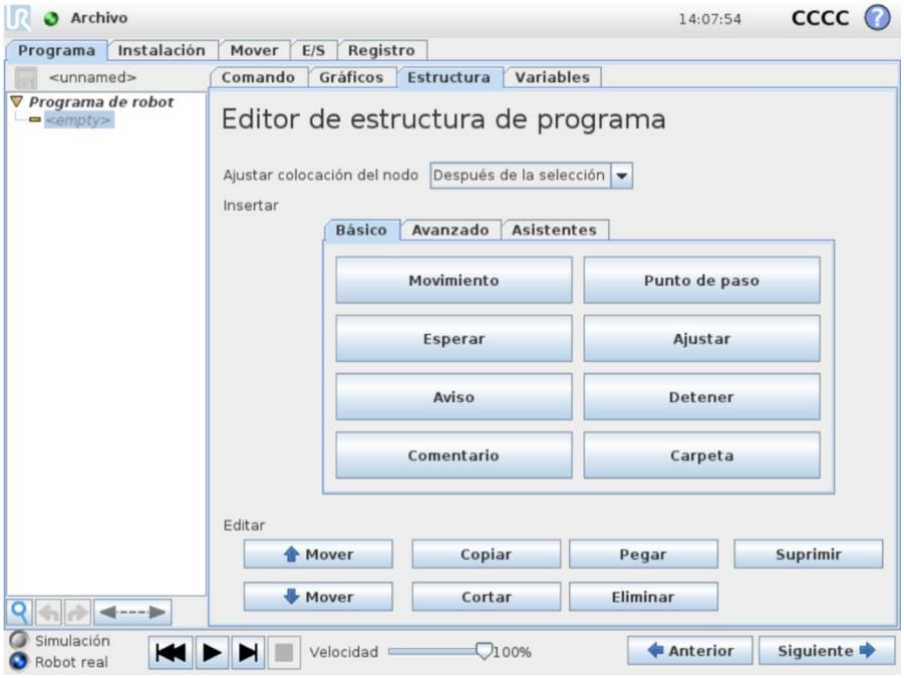

Figura 20.Pantallazo pestaña Estructura. [Fuente: https://www.universal-robots.com/media/207448/ur3_user_manual_es_global.pdf]

En este nivel, se tienen los comandos más básicos para poder crear un programa.

·**Movimiento:** Mediante dicho comando se pueden crear un determinado tipo de movimiento, formado por uno o varios puntos de paso por los que pasará el robot para realizar dicho movimiento.

·**Esperar:** En el comando esperar se tienen varias opciones, esperar una cantidad de segundos que fija el usuario, o bien esperar a la activación o apagado de una señal de entrada digital o señal analógica, o finalmente, esperar a una función que se puede definir.

·**Aviso:** Mediante este comando, se puede programar un aviso durante la ejecución de un programa.

·**Comentario:** Se pueden añadir comentarios en la programación, para explicar o clarificar algunos comandos empleados para la creación de un determinado programa.

·**Punto de paso:** Son los puntos que conforman un movimiento, y por los que el robot ha de pasar.

·**Ajustar:** Este comando permite ajustar el valor de las señales digitales de salida o analógicas de entrada, o bien el valor de las variables que tenga nuestro programa.

·**Detener:** A través de este comando, la ejecución del programa se detiene.

·**Carpeta:** Mediante la carpeta, se puede organizar la estructura del programa para que tenga mayor concisión y claridad.

6.1.5.2. Nivel avanzado

·**Bucle:** Con esta función es posible implementar un bucle siempre, un bucle que se implemente mientras se cumpla una determinada condición o un bucle 'n' veces a determinar por el programador.

·**Asignación:** A través de esta función, se puede asignar un valor a una variable, dicho valor puede ser booleano o un valor numérico, o el valor que se adquiera a través de alguna función del programa.

·**SubProg:** A través del SubProg se pueden crear programas independientes del programa principal y además es una opción muy útil para hacer llamadas dentro del propio programa al subprograma para que se ejecute dentro del programa principal.

·**If…else:** Esta opción implementa una cierta parte del programa si se cumple la condición que está impuesta en el If.

·**Código de script:** El código de script permite ejecutar instrucciones y comandos avanzados, y, por tanto, se puede combinar la programación a través de PolyScope y una programación con instrucciones más avanzadas y complejas.

·**Subproceso:** El subproceso permite implementar un subprograma dentro del propio programa del robot que se ejecuta al mismo tiempo que el programa del robot.

·**Asistentes:** Entre los asistentes que posee PolyScope están la función palé, búsqueda, fuerza y seguimiento de la cinta. Sin embargo, se va a detallar las función de palé y fuerza (aunque no se vaya a utilizar en el proyecto, es interesante comentar la función), ya que la búsqueda y el seguimiento de la cinta no se van a emplear en este proyecto.

·Palé:

Figura 21.Pantallazo función Palé. [Fuente: https://www.universal-robots.com/media/207448/ur3_user_manual_es_global.pdf]

A través de la función palé, es posible realizar la operación de paletizado de una manera muy sencilla.

Como se puede observar en la imagen, se pueden realizar cuatro tipos de patrones: línea, cuadrado, caja o lista.

Se elige el patrón deseado y simplemente se definen los puntos de paso que forman el patrón, un punto de acercamiento, un punto en el cual se deposita la pieza y un punto de salida. Además, después de definir esto, se indica el número de intervalos entre una posición y otra, y ya se tiene definida una función de paletizado.

·Fuerza:

Figura 22.Pantallazo función Fuerza. [Fuente: https://www.universal-robots.com/media/207448/ur3_user_manual_es_global.pdf]

Esta opción permite programar una parte del programa que funcione en modo fuerza. Así pues, se puede una fuerza de tipo simple o compuesta.

·Variables:

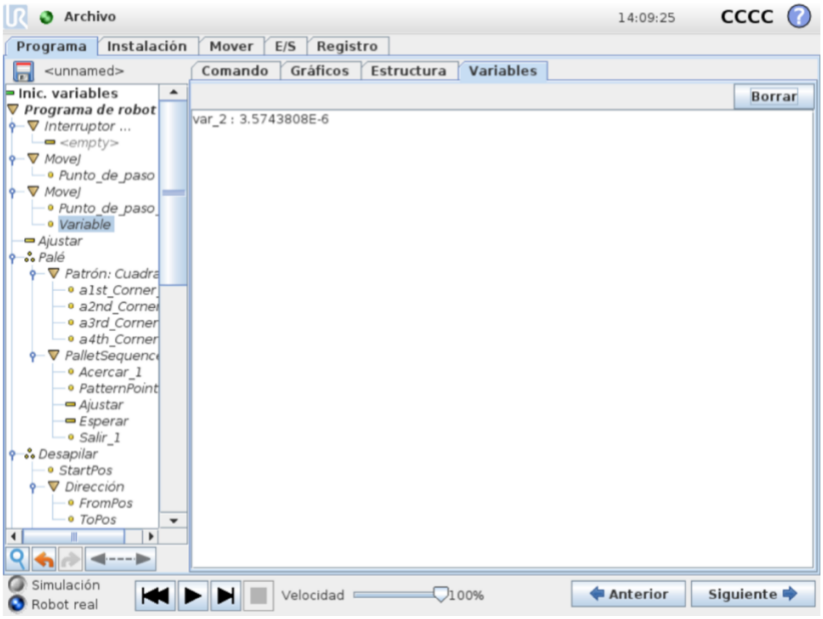

Figura 23.Pantallazo pestaña Variables. [Fuente: https://www.universal-robots.com/media/207448/ur3_user_manual_es_global.pdf]

En esta pantalla, aparecen todas las variables, y sus respectivos valores, definidas en el programa del robot.

·Configuración:

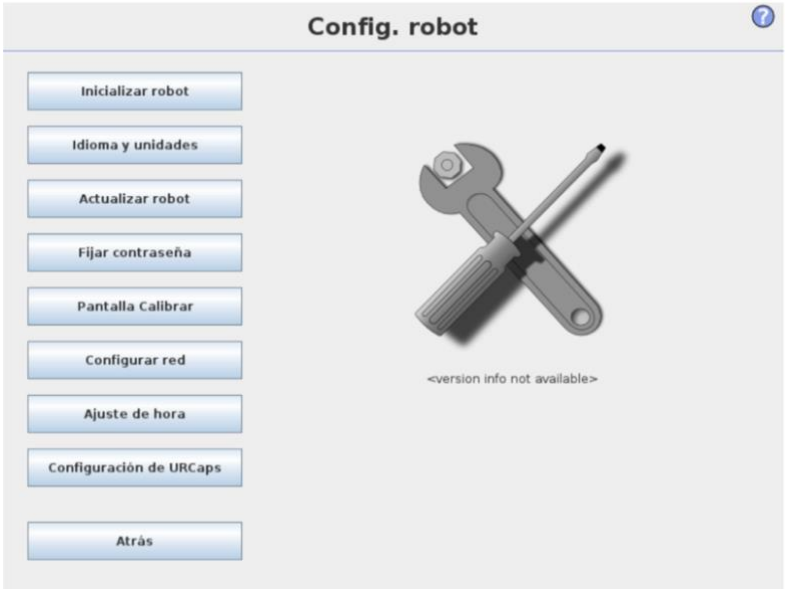

Figura 24.Pantallazo Configuración del robot. [Fuente: https://www.universal-robots.com/media/207448/ur3_user_manual_es_global.pdf]

En esta opción se pueden ajustar parámetros del robot, como cambiar el idioma, actualizar el software del robot, configurar la red, o calibrar la pantalla para facilitar el uso de la interfaz gráfica.

6.2. Programación mediante scripts

Otra forma posible de programar el robot, es haciéndolo mediante scripts a través de lenguajes de programación ya sean, C++, Java o Python entre otros. Este tipo de programación puede ser realizado en Polyscope, añadiendo scripts en el propio programa o también existe la posibilidad de enviar el script a través de un dispositivo externo.

Es posible realizar la programación mediante scripts a través de la comunicación entre el socket propio del robot y un socket externo actuando como servidor, ya sea, por ejemplo, un archivo ejecutable en el ordenador o un dispositivo externo, como por ejemplo, una placa de Arduino.

Así pues, el robot dispone de tres puertos, 30001, 30002 y 30003 donde se pueden enviar los scripts desde un dispositivo externo o desde un servidor. Sin embargo, si se crea una conexión cliente-servidor es mejor utilizar otro número de puerto que los nombrados anteriormente.

En cuanto a las ventajas que ofrece la programación mediante scripts, es destacable la posibilidad de poder editar el programa mientras el robot realiza sus tareas o la posibilidad de introducir y crear nuevas funciones, ampliando las capacidades y posibilidades del robot. Además, también es posible controlar la programación de varios robots al mismo tiempo, a través de un solo ordenador.

Sin embargo, en los siguientes capítulos se verá de forma más detallada la comunicación entre sockets, y las ventajas que supone aplicarlo a los robots colaborativos.

7. INTERFACES HOMBRE-MÁQUINA

Actualmente, es cada vez más interesante poder desarrollar máquinas con las que el ser humano pueda interaccionar, ya que una interacción hombre-máquina, puede facilitar y agilizar muchas tareas y muchas funciones a realizar actualmente en la industria, ya que conforme avanza la sociedad, es necesario que la tecnología avance con ella. Es aquí donde surge el concepto de interfaz hombre-máquina.

Una interfaz hombre-máquina (HMI) es un dispositivo o un sistema mediante el cual se crea una interfaz entre una máquina y el operario, de manera que el operario puede controlar a la máquina en procesos ya sean simples o complejos.

De modo que, como se ha comentado al principio de este trabajo, las dos formas de interfaz hombre-máquina que se van a desarrollar en este proyecto, son por un lado el control por reconocimiento de voz y por el otro, el control por fuerza.

7.1. Reconocimiento de voz

Uno de los puntos más atractivos de la robótica a día de hoy, es la implementación del reconocimiento de voz en el funcionamiento de un robot. Mediante esta funcionalidad, una interfaz natural hombre-máquina cómoda y sencilla, se puede aumentar el rendimiento de cualquier proceso a nivel industrial mediante los robots colaborativos, ya que, como bien se ha detallado anteriormente, estos robots trabajan en conjunción con el operario, facilitando su tarea. Como se puede intuir, con un sistema de reconocimiento de voz, se podría facilitar aun más la tarea del operario, ya que este, simplemente tendría que dar las órdenes correspondientes al robot sin tener que desplazarse ni estar en contacto continuo con él, pudiendo realizar otras tareas a la vez que el robot trabaja.

Uno de los métodos posibles para la implementación del reconocimiento de voz, es mediante placas de Arduino, ya que tienen una relación calidad-precio aceptable, y, por tanto, por un coste muy reducido, y como es en el caso de este proyecto, se puede llevar a cabo el funcionamiento del robot UR3 a través del control por reconocimiento de voz.

Sin embargo, cabe destacar que la placa de Arduino empleada tiene ciertas limitaciones, ya que solo tiene la capacidad de almacenar siete palabras y la resolución de la grabación de dichas palabras es bastante baja, por lo que en ocasiones el robot no obedece a la orden hasta que no se repite dicha palabra varias veces y la placa finalmente detecta el sonido.

7.1.1. Placas de Arduino

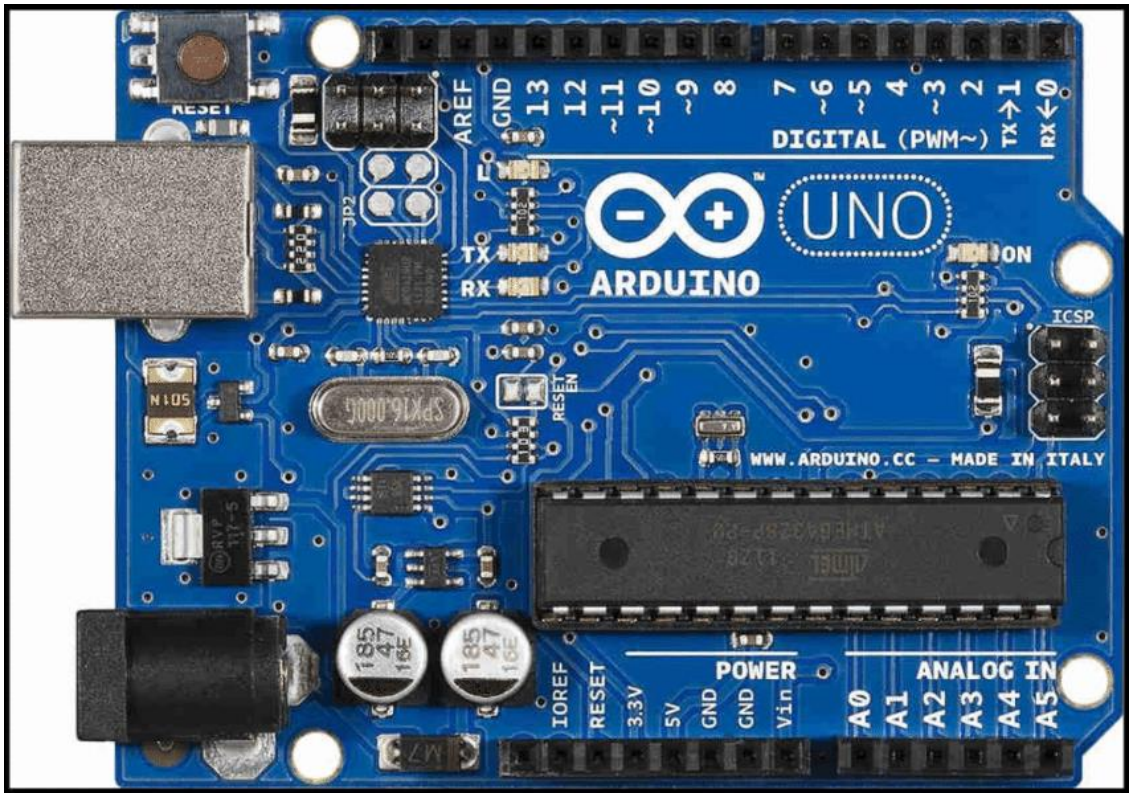

Figura 25. Placa Arduino Uno. [Fuente: https://www.researchgate.net/figure/Arduino-UNO-Board_fig1_320356747]

Arduino es una compañía de creación de hardware y software libre, es decir es de libre utilización y distribución, mediante el cual, la comunidad internacional puede desarrollar un sinfín de aplicaciones diversas. Un ejemplo de aplicación podría ser encender y controlar un motor, o un conjunto de LEDs o como es en este caso, una aplicación que permita controlar un robot colaborativo a través de la voz.

Las ventajas que ofrece Arduino son diversas, y entre las cuales destacan, la posibilidad de la libre utilización y libre distribución, el atractivo precio de los productos Arduino, que las convierte en productos asequibles, la posibilidad de ejecutar su software en Windows, GNU/Linux y Mac OSX (es un software multiplataforma) y el entorno de programación sencillo entre otras de las muchas ventajas que ofrece Arduino.

Como se puede observar en la placa de la Figura 25, es la que se va a utilizar para poder implementar el reconocimiento de voz, posee 6 entradas analógicas(A0-A5) y 14 salidas digitales.

Además, a esta placa se le conecta una placa Arduino Ethernet, para poder realizar la conexión entre el robot y la placa.

7.1.2. Placa Arduino Ethernet

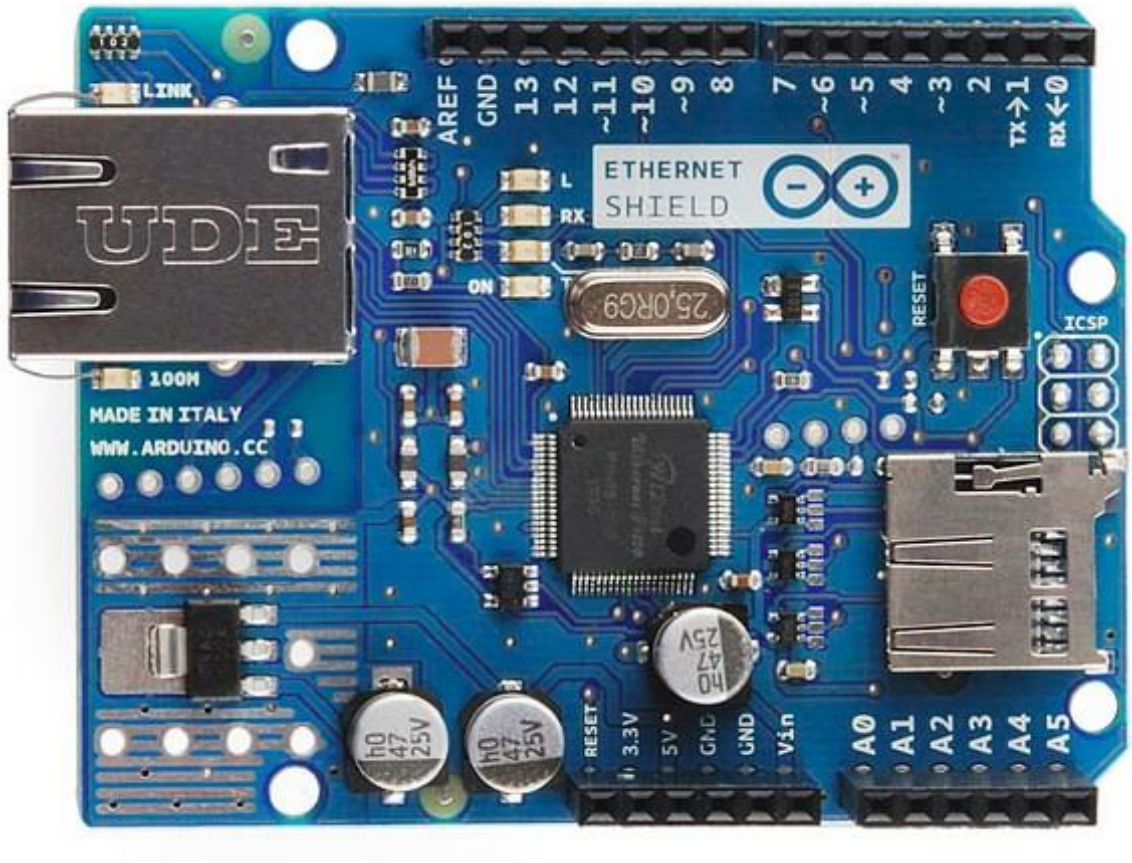

Figura 26. Placa Arduino Ethernet. [Fuente: https://aprendiendoarduino.wordpress.com/2016/07/04/ethernet-shield/]

La placa de la Figura 26 se conecta encima de la placa Arduino (mediante conexión macho-hembra) anterior, y a través de un cable RJ45 se conecta al switch de red donde está también conectado el PLC del robot. Además, también se conecta al ordenador mediante un cable USB conectado a la placa Arduino.

Por último, ya solo será necesario el módulo de reconocimiento de voz que irá conectado a la placa Ethernet.

7.1.3. Módulo de reconocimiento de voz

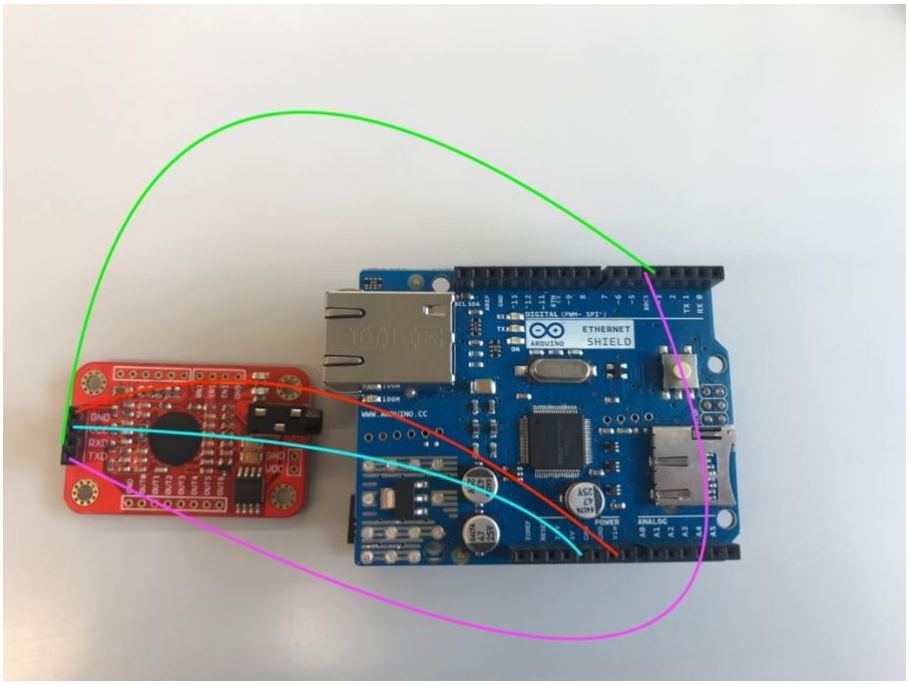

Figura 27. Conexión entre el módulo de reconocimiento de voz y la placa Arduino.

Según la Figura 27, se observa como se conectan el módulo de reconocimiento de voz y la placa Arduino. De modo que, se conectan Rx (color verde) y Tx (color rosa) a los pines 4 y 5 de la placa, la tierra (GND) (color rojo) con la tierra de la placa ethernet y el Vcc (color azul) con los 5V.

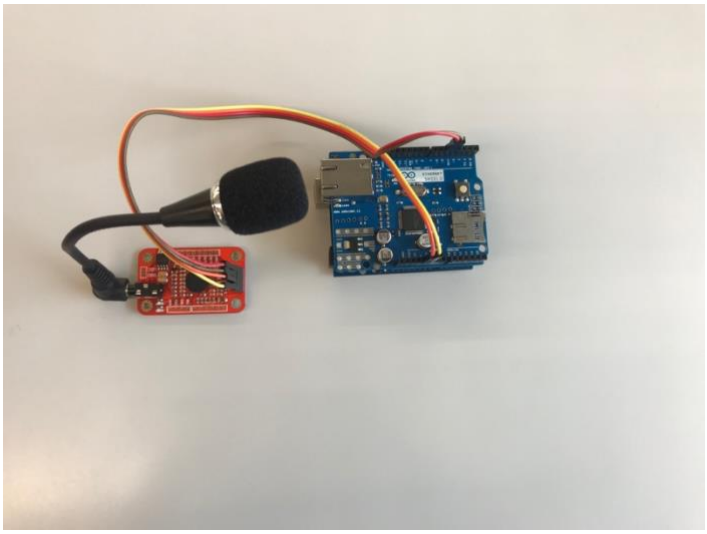

Figura 28. Conjunto placa reconocimiento de voz.

Así, por último, se observa en la Figura 28 que el micrófono se conecta a la entrada que tiene disponible el módulo de reconocimiento de voz, y ya se consigue el sistema necesario para poder realizar el control de voz al robot.

Tras esto, a través de la conexión USB al ordenador, solamente hay que abrir el programa Arduino, abrir y compilar el código programado, y proceder a la grabación de los comandos de voz que va a almacenar la placa. Antes de esto, hay que asegurarse de descargar la librería del módulo de reconocimiento de voz, disponible de manera gratuita en internet.

Por tanto, tras abrir el programa y abrir la librería, se carga, en Archivo→ Ejemplos → VoiceRecognitionMaster-V3 → vr_sample_train

Una vez compilado y subido el módulo, se abre el Monitor Serie y a través del comando sigtrain NÚMERO PALABRA (donde número es el número que se ha asignado a la palabra en el código programado , y la palabra es la que se ha elegido para grabar). Antes de la grabación, se puede observar como la placa tiene parpadeando un LED de color naranja, y tras enviar el comando, deja de parpadear y en ese momento, se ha de decir la palabra, tras lo cual se enciende un LED de color rojo a la vez que el naranja, y se vuelve a repetir el proceso para volver a decir la palabra por segunda vez, y finalmente la placa guarda en su interior la palabra.

Tras esto, se sube el código a la placa, y ya está preparado para controlar el robot.

7.1.3.1. Pseudocódigo de implementación del reconocimiento de voz

A modo de pseudocódigo se va a desarrollar la parte del código que contiene el programa de reconocimiento de voz. A continuación se detallará la funcionalidad de las funciones que lo forman.

Programa: Reconocimiento de voz

Entero a, b, c, d// crear variables

Definición palabra (número asignado). (por ejemplo: libre (4)) //definir los comandos de voz

Inicio del programa

Inicializar servidor // servidor es server

Inicializar servidor 1 //servidor 1 es server1

Inicializar la placa ethernet

Inicializar la configuración de la red

Si cargar comando =>0

Entonces se reconoce el comando de voz

Bucle:

Escucha clientes // Está pendiente de clientes que quieren conectarse a la placa

Entero ret

Guardar comandos grabados en ret

Si ret >0

Entonces:

caso uno:

a= escribir un 1 en el servidor

caso dos:

b= escribir un 2 en el servidor

caso tres:

c= escribir un 3 en el servidor

caso libre:

d= escribir un 4 en el servidor

Fin

Básicamente, el programa consiste en grabar los comandos de voz que se quieran utilizar (máximo 7 comandos) y asignar un número cualquiera a cada comando. Por ello, se crean las variables enteras “a”, “b”, “c” y “d”.

Tras todo el proceso de inicialización de la placa, los servidores y la configuración de red, si se carga un comando (se dice la palabra) y esta función es mayor o igual a 0, entonces el módulo reconoce el comando de voz.

Finalmente, se define una variable denominada ret donde se guardan los comandos grabados y si esta variable es mayor que 0 (es decir, que se ha reconocido la palabra) entonces hay varios casos definidos para cada comando grabado. Así, si se dice, por ejemplo, la palabra “libre” a través del micrófono, el programa entrará en el caso libre y le asignará a la variable “d” escribir el número 4 en el servidor. Si se dice por ejemplo la palabra “uno”, se asignará a la variable “a” escribir el número 1 en el servidor.

7.2. Control de fuerza

El funcionamiento del robot a través del control de fuerza, es una opción interesante en la industria ya que se puede controlar algunas o todas las funciones del robot a través de esta opción de una manera eficaz y rápida, lo que hace más sencillo el modo de funcionamiento del robot.

Este modo de funcionamiento, permite, en un ámbito de trabajo colaborativo, prescindir del uso de la interfaz gráfica del robot para parar el robot, ya que simplemente se necesitaría ejecutar el programa, y a partir de ahí, es donde, a través del sensor de fuerza y aplicando una fuerza, a determinar por el programador, pero en nuestro caso, se utilizará un rango entre 0-85N.

Implementar este control en el robot se lleva a cabo de un modo muy sencillo, ya que simplemente se ha de crear un subprograma en el programa en el robot, donde esté constantemente leyendo el valor de la fuerza aplicado sobre él, y cuando dicho valor de fuerza sea superior a un valor determinado por el operario, una variable llamada Modofuerza se iguala a ella misma pero negada.

Así, se puede, presionando ligeramente sobre el robot, accionar o detener su funcionamiento.

8. SOCKETS

Los sockets son un método de comunicación bidireccional entre varios procesos, que permiten el intercambio de información entre ellos, encontrándose dichos procesos incluso en diferentes máquinas.

Este mecanismo surge en el año 1982, cuando el Grupo de Investigación de Sistemas de Computación de la Universidad de California en Berkeley desarrolla el API de Sockets, publicado además como parte del sistema operativo VSD 4.1c en ese mismo año.

Actualmente, se siguen utilizando los mismos tipos de sockets, ya que el único avance en ellos es que se realizó una extensión para soportar direcciones más largas de IPv6.2.

8.1. Principio de funcionamiento

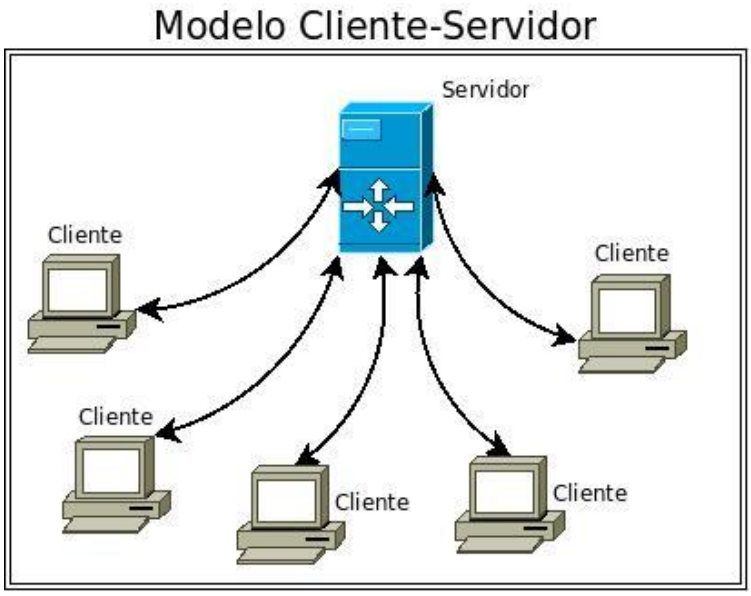

Figura 29. Esquema conexión cliente-servidor.

[Fuente: http://psp.codeandcoke.com/apuntes:sockets]

El principio de funcionamiento de un socket es parecido a la comunicación mediante correo electrónico o la comunicación telefónica, en los cuales se puede intercambiar información aún estando en diferentes lugares. Además, para que se pueda realizar esta comunicación son necesarios una serie de requisitos, tales como, que el cliente pueda localizar al servidor y viceversa y que los datos intercambiados se ajusten a su finalidad.

Para cumplir dichos requisitos, el socket se define a través de direcciones IP local y remota, identificando el origen y el remoto, dos números de puerto local y remoto, para identificar el programa en el computador y un sistema de transporte para el intercambio de datos.

8.2. Comunicación entre sockets

En cuanto a la comunicación entre sockets, el socket es un proceso con un principio de funcionamiento que posee una arquitectura cliente-servidor.

La arquitectura cliente-servidor trata básicamente de un servidor y un cliente. El servidor es un programa en continua ejecución que espera a que el cliente inicie la comunicación con él. De modo que, el cliente es el que solicita la comunicación con el servidor para intercambiar información con él. Una vez establecida la conexión, se le asigna al cliente un número de puerto y comienza la comunicación entre los dos, de manera que, van intercambiándose datos.

8.2.1. Comunicación en el robot UR3

Para establecer la comunicación con el robot UR3 , este tiene un socket que actúa como cliente siempre abierto y preparado para conectarse a un servidor.

Para poder realizar la conexión de la placa Arduino con el robot, el primer paso es conectar a través de un cable RJ45 el robot a un switch de red, y de otro cable RJ45 la placa al mismo switch de red.

En segundo lugar, es necesario incluir en el programa del robot el código que permitirá que dicha conexión se lleve a cabo. Esto se realiza incluyendo un programa antes de empezar la ejecución del programa del robot, y el cual se detallará a continuación.

En tercer lugar, se tiene que crear un subproceso para recibir la información del servidor que se ejecutara al mismo tiempo que el programa del robot, y en el cual se profundizará posteriormente.

9. APLICACIONES DESARROLLADAS

A lo largo de este proyecto, se han desarrollado varias aplicaciones de procesos industriales, para poder demostrar que es posible el control del robot a través del reconocimiento de voz y de la fuerza. Cabe destacar que a lo largo de la programación ha habido ciertas limitaciones, ya que la herramienta que posee el robot (una pinza electro neumática) cuyo agarre de objetos está limitado (ya que no ha sido posible el agarre de objetos con formas esféricas o irregulares, u objetos muy anchos y altos), a diferencia de la ventosa de succión, que permite el agarre de cualquier tipo de objeto con cualquier forma y permite maniobrar con mayor facilidad.

En concreto, cuatro aplicaciones programadas a través de PolyScope son las que forman el proyecto, dos aplicaciones controladas por voz y otras dos controladas por fuerza. Además, una última aplicación que en un futuro próximo se controlará por voz.

Por un lado, las aplicaciones controladas por voz siguen la misma estructura a la hora de la programación. El programa consta de cuatro partes: Inicialización de variables, BeforeStart (antes de empezar), RobotProgram (programa del robot) y un Subproceso.

En primer lugar, en la inicialización de variables, se inicializan todas las variables que se necesiten a un valor determinado por el programador para el correcto funcionamiento del programa. Dependiendo de la aplicación, es imprescindible inicializar las variables cuyo valor es clave, ya que dependiendo del valor que tomen, el robot realizará una acción u otra. De todos modos, en BeforeStart (antes del programa) también se pueden inicializar las variables.

```
BeforeStart
     open:=socket_open("172.16.191.2",90)
     Loop open≟ False
       Wait: 0.01
       open:=socket_open("172.16.191.2",90)
     targetPos:=p[0,0,0,0,0,0]
```

En segundo lugar, en referencia a la parte de BeforeStart (antes del comienzo del programa) hay que destacar ciertos detalles para una mejor comprensión del programa. Se denomina así ya que hasta que no se ejecuten todas las líneas de código que contiene, el programa del robot no puede comenzar. Esta parte es igual en todas las aplicaciones controladas a través de la voz ya que a través de esta parte se consigue la conexión del socket del robot con la placa conjunto que se va a emplear para controlar por voz el robot.

Como se puede observar en el código anterior, solamente hay cinco líneas de código que permiten crear la conexión mencionada anteriormente.

En la primera línea, se asigna a una variable denominada “open” la función “socket_open(IP, puerto) en las cuales IP es la dirección IP de la placa Arduino, en este caso es “172.16.191.2” y el puerto, es el puerto a conectarse de la placa Arduino y se elige entre el puerto 80 y el 90 (definidos en el código de

implementación de reconocimiento de voz), el puerto “90” siendo perfectamente valido el puerto 80 también. A través de esta función el robot se conecta a la placa Arduino.

Es posible que no se consiga la conexión en el primer intento, o que haya algún problema al intentar conectarse. Por ello, se define un bucle en el que mientras la variable “open” sea igual a False entonces hay una pequeña espera de 0.01 segundos y nuevamente se le asigna a la misma variable la función “socket_open(“172.16.191.2”,90). En cuanto a la espera de 0.01 segundos, es importante incluirla dentro del bucle, ya que, si no, el programa detecta un fallo de bucle infinito y no es posible ejecutarlo.

Finalmente, la última línea en la que se le asigna a una variable denominada “targetPos” el valor “p[0,0,0,0,0,0]” mediante el cual se inicializa la tarjeta.

En cuanto al código siguiente, es un subproceso que es igual en todas las aplicaciones de control por voz, exceptuando el número de condiciones if, que dependerá de cuantas órdenes se hayan definido en dicho programa (el número máximo de órdenes puede ser 7). A través de este subproceso se consigue la recepción de datos enviados por el servidor En este caso, se va a detallar cada línea de código que permite esta funcionalidad, asumiendo que solamente hay dos órdenes de voz definidas en la aplicación.

```
Subproceso_1
     receiveFromServ:=socket_read_ascii_float(1)
     'se espera recibir del socket un float entre paréntesis:
"(2.3)", p.e.'
     Loop receiveFromServ[0]≠1
       'si no llega nada, el primer campo ([0]) será 0'
       Wait: 0.1
       receiveFromServ:=socket_read_ascii_float(1)
     If receiveFromServ[1]≟1.0
       esperar:=1
     ElseIf receiveFromServ[1]≟2.0
       esperar:=2
```

En la primera línea se asigna a una variable llamada “receiveFromServ” la función “socket_read_ascii_float(1)" cuya función es leer continuamente los datos enviados del servidor al socket del robot, y el valor 1 indica que solamente se recibe un número. En la siguiente línea hay un comentario que aclara que se espera recibir del servidor un float indicado entre paréntesis.

Tras esto se ha definido un bucle que se cumple mientras el valor de la primera posición de la variable “receiveFromServ” es diferente a 1 (esto indica que hay un error en la recepción del número). Por ello, como pueden ocurrir fallos en la recepción de datos, en el bucle se vuelve a establecer una espera de 0.01 segundos (para evitar el error mencionado anteriormente) y reintentar leer nuevamente los números enviados desde el servidor a través de la función detallada anteriormente. Se seguirá dentro del bucle hasta que se consiga leer correctamente los datos que envía el servidor.

Por último, se definen las condiciones if en las cuales dependiendo del valor que reciba del servidor en la segunda posición de la variable “receiveFromServ” , de ahí que en la condición if se escribe

"receiveFromServ[1]" indicando entre corchetes la posición donde se guarda el dato recibido del servidor. Así, en la primera condición, si el dato de la segunda posición de esta variable es igual a 1.0, entonces se le asigna a una variable denominada "esperar" el valor de 1.

Por tanto, en la segunda condición ocurre lo mismo, pero si se recibe un valor igual a 2.0, entonces la variable "esperar" tomará el valor de 2.

A raíz del valor recibido por el servidor, se observa que se le asigna a una variable un determinado valor para que, en el programa del robot, dependiendo del valor que tenga, en este caso la variable "esperar", el robot realizará una determinada acción u otra.

Por otro lado, las aplicaciones controladas por fuerza siguen todas ellas la misma estructura, sin embargo, esta estructura es más sencilla que la del control por voz. En este caso se prescinde del BeforeStart (antes de empezar) y simplemente el programa se compone de tres partes: Inicialización de variables, RobotProgram (programa del robot) y un Subproceso.

En cuanto a la inicialización de variables, es igual al de los programas controlados por voz, se inicializan los valores de las variables que van a intervenir en el programa.

```
Subproceso_2
  Wait: 0.01
  If force()≥85
    Modofuerza:= not (Modofuerza)
```

Como se puede observar en el anterior fragmento de código, el subproceso mediante el cual se consigue el control por fuerza es muy sencillo, simplemente está compuesto por tres líneas de código.

En primer lugar, a través de la espera de 0.01 segundos se evita el bucle infinito del subproceso. En la siguiente línea, que es una condición if en la cual se indica que si "force()" es mayor o igual a 85N entonces la variable "Modofuerza" pasa a ser ella misma pero negada. A través de la función "force()" el robot está continuamente leyendo el valor de fuerza aplicado sobre él y, por tanto, cuando se presiona ligeramente en cualquiera de las partes del robot y se iguala o sobrepasa el valor, que en este caso se ha establecido 85N, pero podría variarse dicho valor si se deseara.

Así, la variable "Modofuerza" se emplea para el mismo fin que la variable "esperar" , dependiendo del valor que tome esta variable, el robot realizará un movimiento u otro. Por lo general, "Modofuerza" es una variable booleana y, por tanto, puede tomar dos valores, o True o False.

Una vez detalladas las partes que se mantienen igual tanto en las aplicaciones controladas por voz y en las de control por fuerza, se va a detallar cada una de las aplicaciones que forman parte de este proyecto.

9.1. Movimientos accionados a través de control por reconocimiento de voz

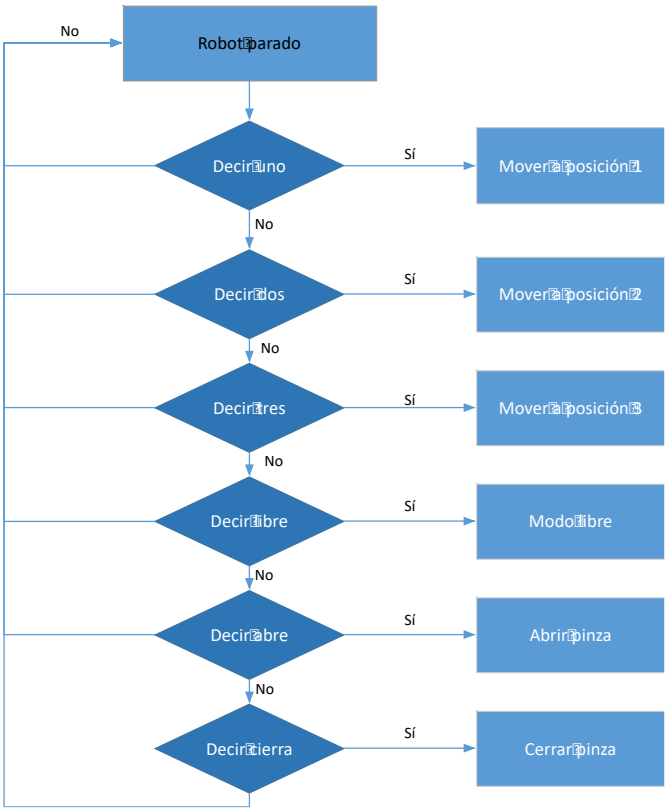

Figura 30. Diagrama de flujos de la aplicación de movimientos accionados a través de la voz.

En esta aplicación se quiere demostrar que es posible un control mediante la voz de los movimientos del robot, e incluso de la herramienta que posee el robot (conectada como una salida digital).

Por un lado, a través del diagrama de flujos de la Figura 30, se ha definido la funcionalidad que tiene el robot en la aplicación. Como se puede observar, la estructura del diagrama es muy sencilla. En un estado inicial, el robot está a la espera de recibir alguna orden definida a través de control por voz, mientras que no reciba ninguna orden válida, el robot seguirá parado. Pueden ocurrir seis casos que dependerán de la orden que se de a través del micrófono.

Así, si se dicen las palabras “uno”, “dos” o “tres”, el robot se moverá a la posición 1, posición 2 o posición 3, respectivamente. Como también se puede ver, si se dice la palabra “libre”, el robot quedará en modo libre (esto quiere decir que deja libres sus ejes) para ser manipulado por el usuario. Por último, si se dicen las palabras “abre” o “cierra”, el robot abrirá o cerrará la pinza respectivamente.

Por otro lado, se va a proceder a detallar la parte técnica de la aplicación, cada una de las líneas de código que conforman el programa y que permiten que el robot realice la función descrita anteriormente.

```
Program

   Init Variables

   BeforeStart
     open:=socket_open("172.16.191.2",90)
     Loop open≟ False
       Wait: 0.01
       open:=socket_open("172.16.191.2",90)
     targetPos:=p[0,0,0,0,0,0]
   Robot Program
     Wait: 0.01
     If esperar≟1
       Wait: 0.01
       end_teach_mode()
       MoveJ
         Punto_de_paso_1
     ElseIf esperar≟2
       Wait: 0.01
       end_teach_mode()
       MoveJ
         Punto_de_paso_2
     ElseIf esperar≟3
       Wait: 0.01
       end_teach_mode()
       MoveJ
         Punto_de_paso_3
     ElseIf esperar≟4
       Wait: 0.01
       freedrive_mode()
     ElseIf esperar≟5
       Wait: 0.01
       Set DO[4]=Off
     ElseIf esperar≟6
       Wait: 0.01
       Set DO[4]=On

   Subproceso_1 // Subproceso para recepción de datos del servidor
     receiveFromServ:=socket_read_ascii_float(1)
     'se espera recibir del socket un float entre paréntesis:
"(2.3)", p.e.'
     Loop receiveFromServ[0]≠1
       'si no llega nada, el primer campo ([0]) será 0'
       Wait: 0.1
```

```
    receiveFromServ:=socket_read_ascii_float(1)
  If receiveFromServ[1]=1.0
    esperar:=1
  ElseIf receiveFromServ[1]=2.0
    esperar:=2
  ElseIf receiveFromServ[1]=3.0
    esperar:=3
  ElseIf receiveFromServ[1]=4.0
    esperar:=4
  ElseIf receiveFromServ[1]=5.0
    esperar:=5
  ElseIf receiveFromServ[1]=6.0
    esperar:=6
```

Se puede observar en el código de programación, una estructura muy sencilla y dividida en cuatro partes (escritas en negrita): Inicialización de variables, BeforeStart (antes de empezar), RobotProgram (programa del robot) y un Subproceso

Profundizando en el RobotProgram (programa del robot), ya que el resto de partes se han detallado anteriormente. Solamente se puede destacar que en este caso en el subproceso a través del cual se consigue la recepción de datos, hay en lugar de dos condiciones if, seis condiciones, ya que en esta aplicación se han utilizado seis diferentes órdenes de voz. Por tanto, la variable “esperar” se define como un entero.

Como en el subproceso se tienen seis condiciones if, la variable “esperar” toma seis valores diferentes, por tanto, en la estructura del programa se han establecido también seis condiciones if, pero en este caso dependen del valor que toma la variable “esperar”, que a su vez se sabe que depende del valor recibido por el servidor. Hay que destacar que inicialmente la variable “esperar” tiene un valor de 0.

En la primera condición en la que, si el valor de la variable “esperar” es igual a 1, entonces el robot se moverá al primer punto de paso (posición 1). Con la segunda condición en la cual la variable “esperar” toma el valor de 2 el robot se moverá al segundo punto de paso (posición 2). Finalmente, con la tercera condición en la cual la variable “esperar” pasa a valer 3, el robot se mueve al tercer punto de paso (posición 3).

Por otro lado, mediante la cuarta condición en la cual la variable “esperar” toma el valor de 4, el robot se libera de sus ejes y entra en modo libre, así el usuario puede manipularlo y moverlo tocándolo simplemente.

En cambio, con la quinta condición en la que “esperar” toma el valor de 5, se abre la pinza, y con la sexta condición en la que “esperar” vale 6, se cierra la pinza.

Volviendo al diagrama de flujos de la Figura 30, es posible observar como “esperar” toma un valor u otro dependiendo de la palabra que se diga. Así, toma el valor de 1 cuando se dice la palabra “uno”, el valor de 2 cuando se dice “dos”, el valor de 3 cuando se dice “tres”, el valor de 4 cuando se dice “libre”,

y finalmente el valor de 5 cuando se dice “abre” y el valor de 6 cuando se dice “cierra”. Estos valores asociados a las palabras se han definido en el código de reconocimiento de voz de la placa Arduino.

En conclusión, se puede observar como este programa, que tiene una estructura muy sencilla y, por tanto, se puede afirmar que es una aplicación sin ninguna complejidad a través de la cual se ha pretendido demostrar como es posible implementar el control por voz a los movimientos del robot sin necesidad de manipular la interfaz gráfica del robot.

Con lo cual, se deja abierta la posibilidad de implementar dicho tipo de control a aplicaciones en el sector industrial que requieran la realización de diferentes acciones o movimientos o accionamiento de salidas digitales que el robot tenga conectadas, según el valor que tome alguna variable definida en el programa, que dependerá de la orden dada por voz. Así, se podría facilitar el funcionamiento del proceso y mejorar los resultados de dicho sector, ya que se disminuiría el tiempo de ejecución del robot y por tanto se reducirán los costes que esto conlleva.

9.2. Función de paletizado accionada a través de control por reconocimiento de voz

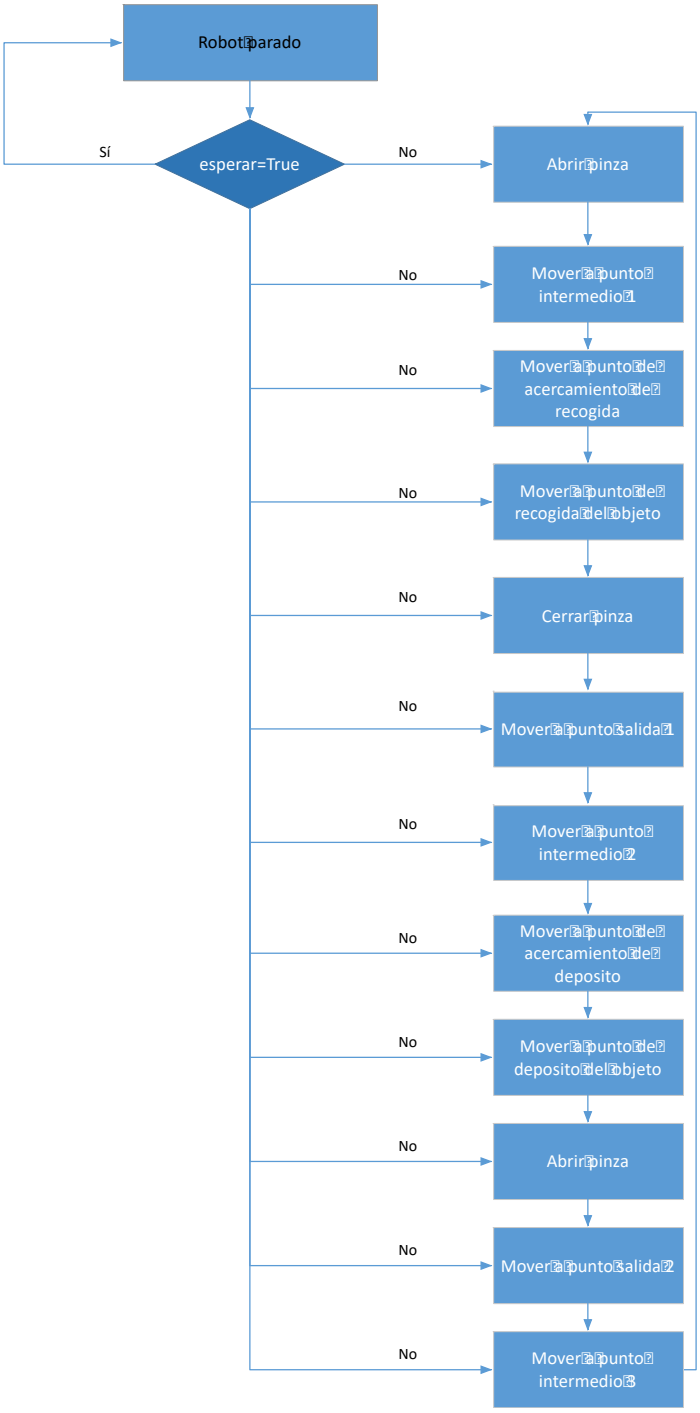

Figura 31. Diagrama de flujos de la aplicación de paletizado accionada a través de la voz.

En este caso, se ha diseñado una aplicación de paletizado para ser controlada a través de la voz. En esta aplicación se ha utilizado un tablero rectangular como palet y se han empleado seis piezas de lego como objetos a paletizar por el robot.

Primeramente, se va a detallar el diagrama de flujos de la Figura 31, para una mejor comprensión de la funcionalidad que tiene el robot en esta aplicación. Se puede observar como, en un estado inicial el robot está parado y esperando a obtener información acerca del valor de la variable definida como “esperar”. Así, si se cumple que “esperar= True”, que quiere decir que todavía no se ha dado ninguna orden o que se ha dado la orden a través de la palabra “para” , el robot seguirá parado. Sin embargo, si no se cumple esta igualdad, quiere decir que se ha dicho a través del micrófono la orden “ya” , por lo que “esperar” pasará a ser False y el robot comenzará el paletizado.

Además, en cualquier punto del programa puede ocurrir que el usuario de la orden de “para”, y entonces en ese momento el robot parará en el punto donde está realizando la acción. Por ejemplo, si está en el punto de “Mover a punto de recogida del objeto” y se da la orden de “para” , el robot no pasará al punto de “Cerrar pinza” , si no que se quedará en dicho punto parado. Para volver a reanudar su movimiento simplemente hay que volver a decir la orden “ya” y el robot continuará su tarea en el punto donde se ha parado (esto quiere decir que no comenzará desde el principio el paletizado, si no que continuará en el punto donde se ha detenido su movimiento).

```
Program

   Init Variables

   BeforeStart
     open:=socket_open("172.16.191.2",90)
     Loop open≟ False
       open:=socket_open("172.16.191.2",90)
     targetPos:=p[0,0,0,0,0,0]
     esperar:= True

   Robot Program
     Loop esperar≟ True
       Wait: 0.1
     Set DO[4]=Off
     Loop esperar≟ True
       Wait: 0.1
     MoveL
       Punto_de_paso_4
     Loop esperar≟ True
       Wait: 0.1
     Pallet // Primer paletizado
       Pattern: Square
         a1st_Corner_2
         a2nd_Corner_2
         a3rd_Corner_2
         a4th_Corner_2
       PalletSequence
         Loop esperar≟ True
```

```
          Wait: 0.1
        Acercar_2
        Loop esperar≟ True
          Wait: 0.1
        PatternPoint_2
        Loop esperar≟ True
          Wait: 0.1
        Set DO[4]=On
        Loop esperar≟ True
          Wait: 0.1
        Wait: 1.0
        Loop esperar≟ True
          Wait: 0.1
        Salir_2
        Loop esperar≟ True
          Wait: 0.1
    MoveJ
      Punto_de_paso_2
    Pallet // Segundo paletizado
      Pattern: Square
        a1st_Corner_1
        a2nd_Corner_1
        a3rd_Corner_1
        a4th_Corner_1
      PalletSequence
        Loop esperar≟ True
          Wait: 0.1
        Acercar_1
        Loop esperar≟ True
          Wait: 0.1
        PatternPoint_1
        Set DO[4]=Off
        Loop esperar≟ True
          Wait: 0.1
        Wait: 1.0
        Salir_1
        Loop esperar≟ True
          Wait: 0.1
    MoveJ
      Punto_de_paso_3

  Subproceso_1 // Subproceso para recepción de datos del servidor
    receiveFromServ:=socket_read_ascii_float(1)
    'se espera recibir del socket un float entre paréntesis:
"(2.3)", p.e.'
```

```
Loop receiveFromServ[0]≠1
  'si no llega nada, el primer campo ([0]) será 0'
  Wait: 0.1
  receiveFromServ:=socket_read_ascii_float(1)
If receiveFromServ[1]≟1.0
  esperar:= False
ElseIf receiveFromServ[1]≟2.0 // Si recibe un 2.0 el robot para
  esperar:= True
```

Por otro lado, hay que ver con más profundidad el código del programa, centrando la atención en RobotProgram (programa del robot). El programa del robot, como se observa, está formado principalmente por dos funciones paletizado, uno para la recogida del objeto, y otro para depositar el objeto en el palet.

Sin embargo, al comienzo del programa, el robot abre la pinza (para asegurarnos de que en el paletizado de recogida de objeto tenga la pinza abierta, para poder coger los objetos) y después se mueve a un punto de paso intermedio (para evitar dislocaciones de alguna parte del robot, y evitar así el paro del programa).

Antes de profundizar más, hay que resaltar un detalle, y es que se puede observar que tanto antes de la abertura de la pinza, antes del movimiento al punto de paso intermedio y antes de entrar en el primer paletizado, se ha definido un bucle en el que mientras se cumpla que la variable “esperar” sea igual a True, hay una espera (por tanto, se consigue parar el robot inmediatamente en el punto donde está, una vez se da la orden de “para”).

Tras esto, comienza el primer paletizado de recogida del objeto. Para definir este palet, se ha utilizado el patrón cuadrado (ya que se han dejado las seis piezas de lego puestas en dos filas y tres columnas, definiendo así un rectángulo). En el patrón cuadrado se ha de marcar las cuatro esquinas que conforman el rectángulo en este caso e indicar el número de intervalos entre esquinas horizontal y verticalmente, que dichos números coinciden con el número de filas y con el número de columnas (en este caso 6 piezas forman dos filas y tres columnas), por tanto, los intervalos serían horizontalmente tres, y verticalmente dos.

Tras definir el patrón, se ha de pasar a la parte de PalletSequence (secuencia del palet) donde hay una serie de puntos de paso (Acercar, PatternPoint y Salir) que hay que definir para que el robot guarde y defina el movimiento que tiene que realizar, y así pueda llevarse a cabo el paletizado de manera correcta.

A través de la siguiente imagen se va a detallar de un modo más preciso los tres puntos anteriores para comprender lo que representa cada punto en el paletizado.

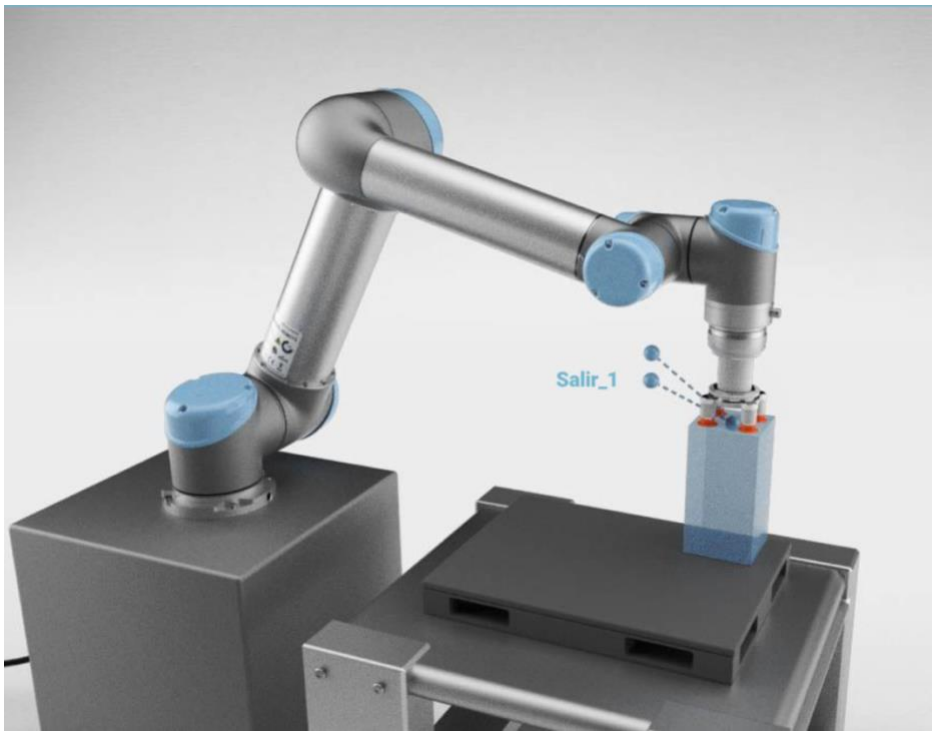

Figura 32. Imagen de los puntos que definen el movimiento de paletizado. [Fuente: https://www.universal-robots.com/es/academy/]

Primeramente, se puede observar en la Figura 32 el punto que está encima del denominado Salir_1. Dicho punto es el denominado como Acercar, el cual se ha de fijar aproximadamente cerca de la primera esquina donde se va a depositar o recoger la primera pieza del paletizado (el caso de la imagen está realizado con depositar un objeto en el palet, pero conceptualmente los puntos representan lo mismo tanto en depositar como en recoger el objeto, por tanto, se definen de igual modo). Después se ha de fijar el PatternPoint, que es el punto donde el robot va a depositar la pieza, y está aproximadamente alineado diagonalmente con el punto Acercar. El último punto en determinarse es el punto de salida, que como se observa en la imagen, está aproximadamente alineado diagonalmente con el punto PatternPoint. Es importante definir estos tres puntos de una manera correcta para evitar dislocaciones en el robot durante el paletizado.

Tras fijar estos puntos, se tiene la secuencia definida, y se pueden añadir más líneas de código que formen parte de la secuencia. En nuestro caso, después del PatternPoint, se añade la opción Ajustar con la que se ajusta el valor de la pinza a HI (se cierra) y así el robot coge la pieza que quiere paletizar.

Una vez fijado el paletizado de recogida, se ha de programar otro paletizado para poder depositar las piezas en el tablero rectangular que se ha simulado como palet. En un principio se había utilizado una caja de cartón que simulaba un palet y era más visible, sin embargo, no era tan estable como el tablero rectangular utilizado finalmente.

Así, para programar el paletizado de deposito del objeto, se sigue un patrón similar al empleado anteriormente (un patrón cuadrado, indicando las cuatro esquinas del tablero) sin embargo, el número de intervalos en este caso horizontalmente serán dos y verticalmente serán tres, ya que las piezas se colocan en tres filas y dos columnas. Además, en este paletizado en la opción Ajustar la pinza, se fija el valor de la pinza a LO (se abre) y así el robot puede depositar la pieza en el tablero. El resto de puntos de PalletSequence (secuencia del palet) se definen del mismo modo que el paletizado anterior.

Tras definir los dos paletizados, el de recogida y el de deposito, se puede observar como se ha definido un movimiento con un punto de paso intermedio a los dos paletizados, para que el robot realice movimientos dentro del alcance de sus ejes y se eviten fallos por dislocación en alguna de sus partes, que harán que el programa se detenga y no funcione correctamente. También, al finalizar el paletizado de depositar la pieza, se ha definido otro punto intermedio con el mismo fin que el punto anterior.

Por otra parte, en cuanto a la manera en la que se controla esta aplicación a través de la voz es muy sencilla y exactamente igual que en la aplicación anterior. Lo único en lo que difiere es que en este caso la variable “esperar” se ha definido como booleana y no como un entero, y por tanto en el subproceso para recibir datos solamente se necesita recibir dos números (para poner la variable “esperar” a True o a False), y, por tanto, solamente hay dos condiciones if.

En la primera condición if, si el servidor envía un 1.0, entonces la variable “esperar” pasa a ser False. En este caso, el 1.0 corresponde con la palabra “ya”.

En la segunda condición if, si el servidor transmite un 2.0, entonces “esperar” toma el valor de True, por tanto, el 2.0 corresponde con la palabra “para”.

Además, hay que observar que el bucle definido en el que mientras la variable “esperar” sea igual a True entonces se cumple una espera de 0.01 segundos, y al ser un bucle, se consigue detener el robot hasta que no cambie el valor de dicha variable. El bucle está puesto antes de cada acción que tenga que realizar el robot dentro de las funciones de paletizado (antes de cada movimiento o de cada accionamiento de la pinza), ya que la función paletizado tiene la limitación de que, hasta que no se realiza toda la secuencia, aunque la variable “esperar” pase a ser True porque se ha dado la orden de “para”, el robot seguiría trabajando y hasta que no completara la secuencia no detendría su funcionamiento.

En resumen, esta aplicación es un proceso muy común en muchos sectores de la industria, ya que el paletizado está fuertemente automatizado, y, por tanto, cabe la posibilidad de implementar el control por voz en procesos de este tipo que pueden ser realizados con robótica colaborativa. El operario puede estar realizando otras tareas en zonas alejadas del robot, y al mismo tiempo, controlar su funcionamiento poniendo en marcha o deteniendo al mismo si necesidad de manipular la interfaz gráfica, y por tanto se aumentaría el rendimiento del proceso.

9.3. Función de paletizado accionada a través de control por fuerza

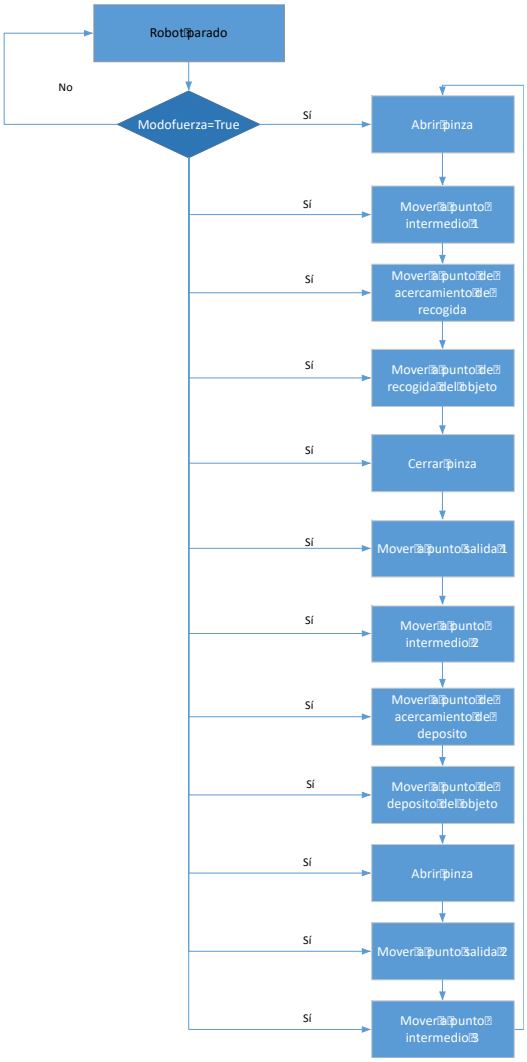

Figura 33. Diagrama de flujos de la aplicación de paletizado accionada a través de la fuerza.

Esta aplicación es el mismo paletizado que la aplicación anterior, pero el modo de controlarlo es diferente, ya que en este caso se controla a través de la fuerza y no a través del control por reconocimiento de voz. Todos los puntos definidos son exactamente iguales a los definidos en la aplicación anterior, y se utilizan las mismas piezas y el mismo tablero para poder realizar el paletizado.

Como se puede observar en el diagrama de flujos de la Figura 33, en este caso también el robot parte de un estado inicial estando parado y esperando ver el valor que toma la variable “Modofuerza”. En este caso “Modofuerza” es una variable booleana que es en un principio está inicializada como False, y una vez se aprieta ligeramente el robot y se cumple la condición del subproceso que lee la fuerza aplicada al robot, pasará a ser True y por tanto se puede observar que el robot comenzará a funcionar. Sin embargo, si se quiere parar el robot, simplemente mientras está funcionando hay que volver a

presionar ligeramente sobre cualquier parte del robot, y entonces la variable “Modofuerza” tomará el valor de False y se parará en el punto donde estaba realizando la acción.

Si se desea volver a reanudar el robot, hay que volver a apretar ligeramente y seguirá su funcionamiento en el punto donde ha parado.

```
Program

   Init Variables

   Robot Program
     Loop esperar≟ True
       Wait: 0.1
     Set DO[4]=Off
     MoveL
       Punto_de_paso_4
     Loop esperar≟ True
       Wait: 0.1
     Pallet // Primer paletizado
       Pattern: Square
         a1st_Corner_2
         a2nd_Corner_2
         a3rd_Corner_2
         a4th_Corner_2
       PalletSequence
         Loop esperar≟ True
           Wait: 0.1
         Acercar_2
         Loop esperar≟ True
           Wait: 0.1
         PatternPoint_2
         Loop esperar≟ True
           Wait: 0.1
         Set DO[4]=On
         Loop esperar≟ True
           Wait: 0.1
         Wait: 1.0
         Loop esperar≟ True
           Wait: 0.1
         Salir_2
         Loop esperar≟ True
           Wait: 0.1

     MoveJ
       Punto_de_paso_2
```

```
    Pallet // Segundo paletizado
      Pattern: Square
        a1st_Corner_1
        a2nd_Corner_1
        a3rd_Corner_1
        a4th_Corner_1
      PalletSequence
        Loop esperar≟ True
          Wait: 0.1
        Acercar_1
        Loop esperar≟ True
          Wait: 0.1
        PatternPoint_1
        Set DO[4]=Off
        Loop esperar≟ True
          Wait: 0.1
        Wait: 1.0
        Salir_1
        Loop esperar≟ True
          Wait: 0.1
    MoveJ
      Punto_de_paso_3

  Subproceso_1 //Subproceso de lectura del valor de fuerza aplicado
    Wait: 0.01
    If force()≥70
      esperar≔ not (esperar)
```

Como se puede observa en el código anterior, en el subproceso que lee la fuerza aplicada en este caso se ha definido que se tiene que cumplir una fuerza mayor o igual a 65N para que “Modofuerza” sea ella misma pero negada.

Así pues, es una manera muy sencilla de controlar una función, que como se ha comentado anteriormente, es tan importante y tan automatizada en el sector industrial. Este modo de controlar permite al operario en aplicaciones de este tipo realizadas con robótica colaborativa, estar cerca del robot realizando otras tareas (en la misma zona de trabajo que él o en zonas aproximadas) y teniendo a su alcance el control del robot de una manera cómoda, rápida y segura.

Por ejemplo, es muy útil cuando el robot termine de paletizar todas las piezas y, por tanto, el palet esté completo y se necesite cambiar. Simplemente con un apriete ligero en la muñeca del robot conseguirá detener el robot y así poder reemplazar el palet completo por uno nuevo y volver a ponerlo en marcha. Así, se evita tener que manipular la interfaz gráfica y por tanto el reemplazamiento de palet se realizará de un modo más rápido y se ahorrará tiempo en la ejecución del proceso.

9.4. Función de paletizado combinada con sensor de presencia y control por fuerza

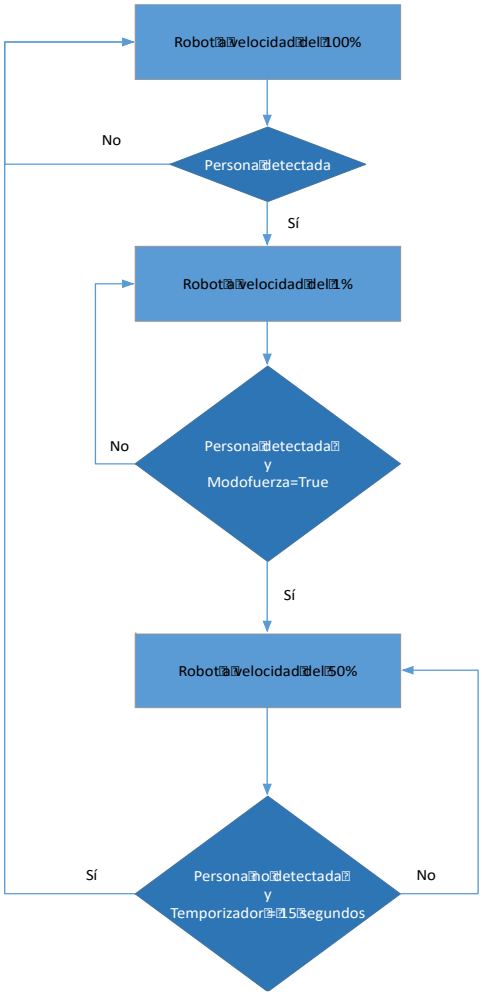

Figura 34. Diagrama de flujos de la variación de velocidad de la aplicación de paletizado combinado con sensor de presencia y control por fuerza.

Esta aplicación es una combinación del control a través de la fuerza y un sensor de presencia instalado en el robot como una entrada digital con dos valores posibles, nivel bajo (LO) o 0 y nivel alto (HI) o 1. Cuando muestra un 0 indica que no hay presencia de personas, en cambio cuando pasa a nivel alto indica presencia de personas. Su modo de funcionamiento es alinear el sensor con un deflector con lo que se consigue crear un haz de luz, y, por tanto, cuando detecta la presencia de una persona es porque se está cortando dicho haz de luz, y se puede observar como se enciende una luz naranja en el sensor como símbolo de que está a nivel alto.

La aplicación consiste en un pick&place simple de cuatro objetos (en este caso, latas de sardina) inicialmente alineadas horizontalmente y tras el pick&place se quedan alineados verticalmente en otra posición.

El funcionamiento es el siguiente:

Observando el diagrama de flujos de la Figura 34 , se puede ver como el proceso ocurre a velocidad normal (del 100%) mientras no haya presencia de personas (el sensor de presencia está a nivel bajo). En el momento en el que se detecta una persona, inmediatamente el robot pasa a funcionar con una velocidad del 1% (prácticamente nula, no se podía pasar al 0% ya que la barra de control de velocidad tiene un rango de 1-100%).

Con el robot prácticamente parado, si la persona sigue en la zona (sigue siendo detectada) y además aprieta ligeramente al robot en cualquiera de sus partes (por tanto la variable “Modofuerza”, que inicialmente está a False, toma el valor de True, es el mismo funcionamiento que el detallado en la aplicación anterior), el robot pasará a funcionar a una velocidad reducida del 50% (para evitar daños a la persona que está en la zona de trabajo del robot y que el robot pueda seguir realizando su tarea).

Finalmente, si no se detecta a nadie, comienza el funcionamiento de un temporizador y cuando pasan 15 segundos, entonces el robot volverá a funcionar al 100% (ya que se asegura que no hay personas a su alrededor). Si esto no ocurre, seguirá funcionando a una velocidad reducida del 50%.

```
Program

   Init Variables

   BeforeStart
     Stop_Timer:= True
     Timer:=0

   Robot Program
     MoveJ
       Punto_de_paso_1
       Set DO[4]=Off
       Pallet
         Pattern: Line
           StartPos_1
           EndPos_1
         PalletSequence
           Acercar_1
           PatternPoint_1
           Set DO[4]=On
           Wait: 1.0
           Salir_1
       MoveJ
         Punto_de_paso_2
       Pallet
         Pattern: Line
           StartPos_2
           EndPos_2
         PalletSequence
```

```
        Acercar_2
        PatternPoint_2
        Set DO[4]=Off
        Wait: 2.0
        Salir_2


Subproceso_1 // Subproceso de variación de velocidad
  ok:=socket_open("127.0.0.1",30002)
  vel_nula:=0.01
  vel_reduc:=0.5
  vel_alta:=1.0
  Wait: 0.01
  Loop
    Wait: 0.01
    Wait: 0.01
    If digital_in[3]≟ True
      Timer:=0
      speed:=vel_nula
    Wait: 0.01
    If digital_in[3]≟ True  and Modofuerza≟ True
      speed:=vel_reduc
    If digital_in[3]≟ False  and Timer≥1500
      speed:=vel_alta
      Modofuerza:= False
    Wait: 0.01
    socket_send_string("set speed")
    socket_send_string(speed)
    socket_send_byte(10)

SubProgram_1 //Subprograma de temporizador
  Wait: 1.0
  Stop_Timer:= False
  Wait: 15.0
  Stop_Timer:= True

Subproceso_2 //Subproceso de lectura de la fuerza aplicada
  Wait: 0.01
  If force()≥85
    Modofuerza:= not (Modofuerza)

Subproceso_3 //Subproceso del temporizador
  If Stop_Timer≟ False
    Timer:=Timer+1
  Wait: 0.01
```

```
Subproceso_4
  Wait: 0.01
  If digital_in[3]≟ False  and speed≟vel_reduc
    Call SubProgram_1
```

En cuanto a la parte técnica, observando el código se ve como el RobotProgram(programa del robot) está compuesto por dos paletizados (uno de recogida del objeto y otro de deposito del objeto) definidos de la misma forma que los paletizados de las aplicaciones anteriores.

Además, también hay dos puntos de paso intermedios para evitar dislocaciones, y al principio del programa se abre la pinza para asegurarse de que está abierta cuando el robot recoja el objeto.

Por un lado, para conseguir la reducción de la velocidad, se puede observar en el código que se utiliza el Subproceso_1. En este subproceso, en la primera línea se asigna a una variable denominada "ok" la función "socket_open(IP, puerto)" donde IP en este caso es 127.0.0.1 que es la IP del robot (se conecta a sí mismo), y el puerto a conectarse del robot para poder manipular la velocidad es el 30002.

En las tres líneas siguientes, se asigna a las variables "vel_nula" , "vel_reduc" y "vel_alta" los valores de 0.01 (1%) , 0.5 (50%) y 1.0 (100%) respectivamente. Tras esto, se define un bucle que se cumple siempre y dentro del bucle ocurren los tres casos que pueden ocurrir para reducir o aumentar la velocidad.

En la primera condición if si se cumple que "digital_in[3]" sea igual a True , siendo "digital_in[3]" la entrada donde está conectada el sensor de presencia, que esto quiere decir que se detecta a una persona, entonces la variable "speed" se iguala a "vel_nula" (funcionará al 1%).

En la segunda condición if, si se cumple que "digital_in[3]" sea igual a True y además que "Modofuerza" sea igual a True (que se haya apretado el robot) entonces "speed" pasará a valer "vel_reduc" (funcionará al 50%).

Finalmente, en la tercera condición if si "digital_in[3]" es igual a False y además el valor del temporizador es de 15 segundos "Timer=1500" (1500 centisegundos) entonces "speed" será igual a "vel_alta" (funcionará al 100%).

Tras esto, se observan tres líneas de código de script. La primera y segunda línea es la función "socket_send_string(cadena a enviar)" a través de la cual se envían las cadenas "set speed" y speed, la primera para regular la velocidad, y la segunda cadena, para enviar la variable speed. Por último, se envía a través de la función "socket_send_byte(10)" el valor de 10, que significa pasar a una nueva línea, para evitar que se sobrescriban en una misma línea los valores enviados.

Por otro lado, para llevar a cabo el control de la fuerza, se utiliza el mismo subproceso (Subproceso_2) utilizado en las aplicaciones anteriores, pero en este caso el valor de fuerza establecido es de 85N (ya que así se evita que se active "Modofuerza" a True debido a las inercias del robot al realizar el pick&place.

Finalmente, para el temporizador se observa en el código que hay un subprograma (Subprogram 1) en la cual se asigna a una variable definía como "Stop_Timer" el valor de False, y tras una espera de 15 segundos, la variable "Stop_Timer" pasará a ser True y por tanto se detendrá el temporizador.

Después, en el Subproceso_4, cuando se cumple que el sensor no detecta a nadie y el robot funciona a velocidad reducida, entra dentro del if donde se llama al subprograma creado para el temporizador. Paralelamente a esto, en el Subproceso_3, mientras la variable "Stop_Timer" sea igual a False, se aumenta en 1 la variable "Timer" (que es donde se almacena el tiempo transcurrido) y posteriormente se pone una espera de 0.01 segundos (simulando un reloj).

Hay que observar como en el Subproceso_1 (para el control de la velocidad) en la primera condición, se le asigna a la variable "Timer" el valor de 0 ya que una vez vuelve a funcionar a la velocidad de 100% se ha de inicializar ese valor para que el programa siga funcionando correctamente puesto que si vuelve a reducirse la velocidad y no hay presencia de personas, el temporizador no comenzaría a contar desde 0, si no que lo haría desde 1500. También en la tercera condición la variable "Modofuerza" pasa a ser False por el mismo motivo, se inicializa para que cuando vuelva a una velocidad del 100% no tenga problemas en reducir y aumentar la velocidad si se detecta presencia de personas, ya que si no se inicializara a False, en el momento en el que el sensor detectara una persona, no funcionaría a velocidad del 1%, si no que pasaría a funcionar a velocidad del 50% (porque "Modofuerza" sería igual a True y entraría en la segunda condición if).

A través de esta aplicación se consigue controlar la velocidad del robot y asegurar la seguridad del operario cuando tenga que estar en la zona de trabajo del robot. Es una opción que podría implementarse en aplicaciones en las que el robot colaborativo pueda suponer cierto peligro para el operario y poder controlar la velocidad a través del valor que tomen ciertas entradas o salidas digitales del robot.

9.5. Aplicación de fresado realizado con un UR10

Durante la realización de prácticas en la empresa ISTOBAL S.A. , la autora de este proyecto ha realizado un programa para mecanizado de cajas. ISTOBAL S.A es una empresa pionera en el sector de lavado de automóviles, vehículos grandes como camiones o buses e incluso lavado de trenes. En los diferentes tipos de maquinaria que fabrican, como puentes, centros o túneles de lavado hay una parte eléctrica, y es aquí donde intervienen las cajas que se mecanizan con el UR10. Estas cajas de diversos materiales, como, por ejemplo, fibra de vidrio, se utilizan como cajas de conexiones, una vez se monta en su interior los componentes eléctricos.

Según un estudio realizado en la empresa, la mecanización manual de estas cajas por operarios suele tener una duración de aproximadamente 50 minutos, en cambio, con el robot se consiguen mecanizar en aproximadamente 7 minutos (con variaciones dependiendo del tipo de caja y de los agujeros que se necesiten mecanizar). Además, también se consigue un ahorro considerable en coste de mano de obra, según cálculos aproximados de 12000 €/año.

Para poder llevar a cabo el mecanizado en el robot se ha instalado una fresadora fijada como herramienta y, además, tiene integrada en la herramienta un sistema de aspiración con la que se aspiran todas las virutas que va arrancando de la caja mecanizada. Además, dicho robot se encuentra en una especie de habitación-jaula, donde hay dos sensores finales de carrera conectados como entradas digitales, uno en la puerta y otro en una ventana), de modo que si la puerta y la ventana no están correctamente cerradas el robot no ejecuta el programa ya que interponerse en su zona de trabajo puede ser peligroso para el operario si la fresadora está encendida.

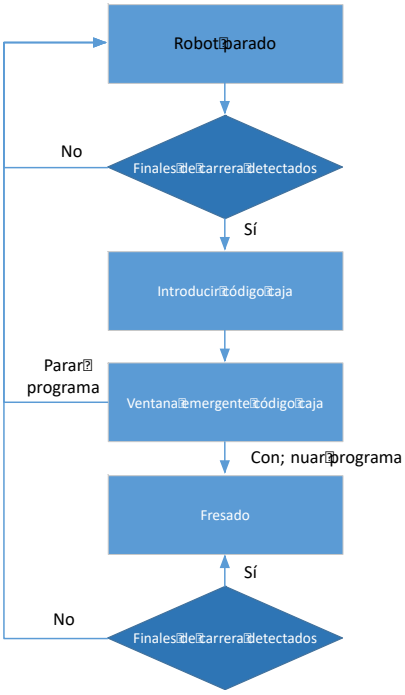

Figura 35. Diagrama de flujos de la aplicación de fresado con un UR10

Como se observa en el diagrama de flujos de la Figura 35, el robot inicialmente está parado, y cuando se ejecuta, si los sensores de final de carrera detectan a estos, entonces sale una ventana emergente en la que el operario tiene que introducir el código de la caja a mecanizar, por el contrario, el robot permanecerá parado. Una vez introducido, simplemente tiene que aceptar y posteriormente aparece otra ventana emergente con el código de la caja que se va a mecanizar y con dos opciones: Parar programa o Continuar. Se pulsa en Continuar y el robot ya comienza el mecanizado de la caja en cuestión. En el momento en el que un sensor deje de detectar un final de carrera, automáticamente el robot se parará en el punto donde estaba funcionando hasta que no se vuelva a detectar dicho final de carrera.

Por otro lado, la parte técnica del robot está compuesta primeramente por condiciones if en las que dependiendo el código de caja que se introduzca, el robot entrará en un determinado if u en otro. Actualmente solamente se disponen de nueve condiciones if, ya que solamente se ha realizado el programa de 9 tipos de caja diferentes. Además, también hay una serie subprogramas para realizar el mecanizado de agujeros con variedad de diámetros, por ejemplo, un subprograma para realizar agujeros laterales izquierdos de diámetro 5mm, otro subprograma para realizar agujeros laterales derechos de diámetro 12mm u otro subprograma para realizar agujeros frontales de diámetro 17mm, etc.

Estos subprogramas se han creado ya que, dependiendo de la caja, puede necesitar agujeros de diferentes diámetros en la cara frontal, en la cara lateral izquierda o según se especifique en el plano de la caja. Por ello, para cada caja hay que realizar un programa nuevo con diferentes puntos de paso.

En cuanto lo que se pretende con esta aplicación, es realizar a cabo pruebas con el robot para intentar llevar a cabo una interacción hombre-máquina, mediante el control de esta aplicación a través de la voz o cualquier otro método visto en este proyecto, sin necesidad de tener que teclear en la interfaz gráfica el código de la caja, simplemente con una orden de voz indicarle al robot la caja que se desea mecanizar u otra orden para poder parar el robot en cualquier momento. La finalidad es facilitar la tarea al operario y que utilizar el robot no sea algo complejo y peligroso, si no, algo accesible, cotidiano y cómodo para él.

Además, para intentar que la interacción con el robot no sea peligrosa sin necesidad de tener que "encerrarlo" en una especie de habitación-jaula sería una opción interesante reemplazar dicha habitación por sensores de presencia alrededor de la zona de trabajo del robot.

A través de estos sensores de presencia se podría realizar un control similar al de la aplicación anterior, donde cuando se detecta una persona, el robot inmediatamente reduce su velocidad al 1% (prácticamente nula), por lo que no habría ningún peligro para el operario estar en la zona de trabajo del robot y se ahorraría costes y también espacio en la fábrica, ya que dicha habitación-jaula tiene unas dimensiones considerables.

Finalmente, son opciones que están en una fase prematura de estudio, pero esta aplicación es un proceso real que se lleva a cabo dentro de una fábrica y que deja abierta la posibilidad a ser mejorada aplicando los controles vistos en este proyecto. Cabe destacar que el control por voz no se ha podido probar con la placa Arduino que se ha utilizado ya que al ser la fábrica un ambiente ruidoso, no es viable realizarlo con esta placa. Así, posteriormente a este proyecto, se va a buscar otras placas con una precisión más exacta y capaces de filtrar el ruido para poder llevar a cabo el control por voz de una manera efectiva.

10. CONCLUSIONES

El fin de este trabajo es profundizar e implementar dos formas de controlar novedosas y sencillas de robots colaborativos. A medida que avanza la sociedad, avanza también la industria y, por tanto, la robótica y la automatización ha de seguir desarrollándose para suplir las necesidades que tiene la sociedad. Así pues, la robótica colaborativa es una herramienta fácil, rápida, y sobretodo muy potente para poder desarrollar más cualquier tipo de proceso industrial susceptible a automatización. Pero también, es necesario avanzar en la forma de controlar dichos robots, y, por tanto, centrarse en el desarrollo de estas dos formas de controlar puede facilitar mucho dicho control.

En cuanto al control a través de la voz de robots colaborativos, es un método sencillo de controlar el robot y cuya implementación en procesos industriales podría aportar mejoras de rendimiento, que se traducen en aumento de beneficios en una determinada empresa. Sin embargo, para poder implementar dicho control en una fábrica, es necesario disponer de un sistema con una gran resolución (que no detecte el ruido que pueda haber en un entorno ruidoso como es una fábrica). Así, sería interesante profundizar en el desarrollo de sistemas de control más potentes, ya que, la placa mediante la cual se ha realizado el control tiene limitaciones en cuanto a número de comandos grabados y en ocasiones, al reconocer una orden de voz dada.

Además, cada vez que se quiera realizar el control, se necesitaba volver a grabar los comandos en el ambiente donde se encuentra el robot, puesto que la placa no es capaz de filtrar los ruidos que puede haber en la zona desde donde se controla el robot. De modo que, sería interesante investigar en algún método para la filtración de ruidos, y así aumentar la precisión del reconocimiento de voz sin tener la necesidad de graba cada uno de los comandos cada vez que se haya de controlar el robot a través de dicho método.

Otra de las limitaciones que tiene el control por voz de este proyecto, es que se ha de realizar desde una zona fija, ya que el conjunto está conectado a un ordenador por un cable USB y a un switch de red por un cable RJ45 y no es posible desplazar dicho conjunto, o desplazar el micrófono por el que se ha de hablar. Una opción futura que podría investigarse sería un método mediante el cual, el usuario podría controlar el funcionamiento desde cualquier lugar (con un cierto alcance), y no tener que estar en una posición determinada para ello. Esto se podría estudiar viéndose si es posible, a través de un micrófono inalámbrico que disponga de sistema Bluetooth, conectarlo a una placa que disponga del mismo sistema y poder así disponer de un control inalámbrico.

Además, una opción podría ser también intentar fusionar el control inalámbrico con una filtración del ruido que pueda existir en el ambiente

En cuanto al control por fuerza, es una opción que no tiene tantas limitaciones como el control por voz.

Sin embargo, sería interesante desarrollar más esta forma de control, aumentando, por ejemplo, la capacidad de los sensores de fuerza que dispone el UR3 y así conseguir sensores más precisos a la hora de detectar la fuerza aplicada.

También, esta forma de control tiene la limitación de que no se puede utilizar en procesos peligrosos los que el robot colaborativo se utilice, como, por ejemplo, una aplicación de fresado en el cual la fresa es una herramienta muy peligrosa para el ser humano si está en funcionamiento y se podría poner en peligro la integridad del operario.

En definitiva, estas formas de control de un robot colaborativo acercan al ser humano con la robótica, ya que son interfaces naturales hombre-máquina, en las que el ser humano realiza acciones naturales como es el hablar o es interaccionar a través del tacto con el robot, sin necesidad de procesos ni métodos complejos para controlar una máquina tan compleja como puede ser un brazo robótico. Sin embargo, son métodos de control que podría decirse que todavía se encuentran en una fase prematura para la robótica colaborativa y es posible investigar más sobre ellas y seguir desarrollando y potenciando sus capacidades.

11. BIBLIOGRAFÍA

Enlaces web consultados

- https://www.universal-robots.com/es/acerca-de-universal-robots/noticias/historia-de-los-cobots/
- http://wiki.robotica.webs.upv.es/wiki-de-robotica/introduccion/clasificacion-de-robots/
- https://dosideas.com/noticias/actualidad/574-pasado-presente-y-futuro-de-los-sockets
- https://www.ecured.cu/Socket
- http://www.roboticaparatodos.es/tipos-de-robots-segun-su-cronologia/
- http://conozcamoslarobotica.blogspot.com/p/generaciones-de-la-robotica.html
- http://www.zacobria.com/universal-robots-knowledge-base-tech-support-forum-hints-tips/universal-robots-script-programming/
- https://new.abb.com/products/robotics/es/robots-industriales/yumi
- http://www.elmundo.es/economia/2015/11/27/5658a159268e3e3f618b4594.html
- https://www.kuka.com/es-es/productos-servicios/sistemas-de-robot/robot-industrial/lbr-iiwa
- http://www.zacobria.com/universal-robots-knowledge-base-tech-support-forum-hints-tips/universal-robots-script-programming/
- http://www.zacobria.com/universal-robots-knowledge-base-tech-support-forum-hints-tips/knowledge-base/script-client-server/
- https://www.universal-robots.com/how-tos-and-faqs/how-to/ur-how-tos/setting-the-speed-slider-from-a-program-15293/
- https://www.universal-robots.com/how-tos-and-faqs/how-to/ur-how-tos/design-a-timer-function-15545/
- https://www.wonderware.com/es-es/hmi-scada/what-is-hmi/
- http://www.masingenieros.com/portfolio/el-nuevo-reto-la-industria-4-0/
- http://sopa.dis.ulpgc.es/ii-dso/leclinux/ipc/sockets/sockets.pdf

Libros consultados:

- Barrientos, Antonio; Peñín, Luis Felipe; Balaguer, Carlos; Aracil, Rafael (Ed. McGraw-Hill). (2007). Fundamentos de robótica
- Rentería, Arantxa; Rivas, María (Ed. McGraw-Hill). (2000). Robótica industrial: Fundamentos y aplicaciones
- Mellado Arteche, Martín (Ed. UPV). (2009). Robótica

Documentos consultados:

- *Manual de usuario UR3/CB3.[Fuente: https://www.universal-robots.com/media/207448/ur3_user_manual_es_global.pdf]*
- *The URScript Programming Language. Version 3.3.4 December 8, 2016. [Fuente: https://s3-eu-west-1.amazonaws.com/ur-support-site/22198/scriptManual.pdf]*

PRESUPUESTOS

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

1. PRESUPUESTOS

1.1. INTRODUCCIÓN

En este apartado se detallarán y se calcularán los presupuestos de este proyecto. De modo que se realizarán los presupuestos diferenciando los costes de la mano de obra y los costes de material utilizado para la realización del proyecto.

En este presente Trabajo de Fin de Grado (TFG), se han dedicado 10 semanas con una dedicación de 6 horas diarias 5 días a la semana, por lo que en total se han dedicado 300 horas a la realización de este proyecto.

Por tanto, en los siguientes apartados se detallarán los costes unitarios que hay que tener en cuenta para el cálculo del coste total

1.2. MANO DE OBRA

En el proyecto hay que tener en cuenta un tipo de mano de obra:

-Graduado en Ingeniería en Tecnologías Industriales con un coste unitario de 40€/h

Tabla 1. Coste unitario de la mano de obra.

Concepto	Coste unitario (€/h)
Graduado/a en Ingeniería en Tecnologías Industriales	**40**

1.3. MATERIALES

Los materiales utilizados en la realización del proyecto se detallan en la tabla de a continuación, donde se detalla el precio, el rendimiento(o vida útil del material) y el coste unitario que suponen en el proyecto.

- Robot UR3: Se ha utilizado el robot UR3 para la programación de las aplicaciones. El coste del robot es de 20000€ con una vida útil de aproximadamente 35000 horas.
- Ordenador personal(MacBook Pro): Se ha empleado el ordenador personal, un ordenador portátil, para búsqueda de información para realizar el proyecto, para programar el código que permite la implementación del control por reconocimiento de voz (a través del programa Arduino) y finalmente para la realización. El coste del ordenador es de 1500 euros, con una vida útil estimada de 8800h. No se tiene en cuenta la devaluación del precio conforme la evolución del tiempo.

- Torre y monitor ordenador laboratorio : Este conjunto se ha empleado puesto que en el ordenador del laboratorio se tenía un simulador en forma de ejecutable que se ha utilizado para simular el control por reconocimiento de voz, y controlar el robot por dicho método. Además, también se ha utilizado para consultar información por internet y consultar manuales de programación del robot. Se estima un coste aproximado de 700€ para la torre y un coste de 200€ para el monitor. También estimamos una vida útil aproximada de 5400h para ambos.
- Placa Arduino: La placa Arduino tiene un coste aproximado de 15€ sin tener en cuenta la amortización y la devaluación del precio con el tiempo.
- Placa Arduino Ethernet: La placa Arduino Ethernet tiene un coste aproximado de 25€ sin tener en cuenta la amortización y la devaluación del precio con el tiempo.
- Módulo reconocimiento voz: Este módulo se ha utilizado para conectarse a la placa Arduino, y está compuesto de una placa y un micrófono. Se supone un coste total de 17.5€ sin tener en cuenta la amortización y la devaluación del precio con el tiempo.
- Cable USB: Se ha empleado un cable USB para conectar el conjunto de la placa Arduino y el módulo de reconocimiento de voz al ordenador y poder cargar el código de implementación Arduino a la placa. Tiene un coste aproximado de 2€.
- Cable RJ45: Este cable se ha empleado para conectar el conjunto de la placa Arduino y el módulo de reconocimiento de voz al switch de red donde está conectado el PLC del robot. Tiene un coste de aproximadamente 3€.
- Switch de red de 8 puertos: Tiene un coste aproximado de 30€.
- Microsoft Office Professional 2016: A través de esta herramienta se ha redactado la memoria y realizado la presentación del trabajo. Su licencia tiene un coste de 69€ y la amortización se estima de aproximadamente 1825h.
- Reprografía: La impresión de la memoria de este proyecto se fija en un coste aproximado de 50€, pudiendo finalmente emplearse menos cantidad para este fin.

Por tanto, para el cálculo del coste unitario se utiliza la siguiente relación:

$$\text{Coste unitario}(€/h) = \frac{Precio(€)}{Rendimiento} \qquad (1)$$

Tabla 2. Costes unitarios de los materiales empleados.

Concepto	Precio(€)	Rendimiento	Coste unitario(€/h)
Robot UR3	20000	35000	**0.57**
Ordenador personal(MacBook Pro)	1500	8800	**0.170**
Torre ordenador laboratorio	700	5400	**0.13**
Monitor ordenador laboratorio	200	5400	**0.037**
Placa Arduino Uno	15	N/A	**N/A**
Placa Arduino Ethernet	25	N/A	**N/A**
Módulo reconocimiento voz	17.5	N/A	**N/A**
Cable USB	2	N/A	**N/A**
Cable RJ45	3	N/A	**N/A**
Switch de red de 8 puertos	30	N/A	**N/A**
Microsoft Office Professional 2016	69	1825	**0.038**
Reprografía	50	N/A	**N/A**

1.4. PRESUPUESTO GENERAL

1.4.1. Partida 1: Mano de obra

Previamente a la tabla con los costes, se va a profundizar en el modo de distribuir el tiempo:

-Realización del trabajo: El desarrollo del proyecto tiene varias fases. Por un lado, la programación del código para implementar el control por reconocimiento de voz, la programación de las aplicaciones del robot y la redacción de la memoria. En cuanto a las horas dedicadas al robot, se pueden estimar aproximadamente en 120h de trabajo en el laboratorio, y en cuanto a la programación del código Arduino y la redacción de la memoria se fija aproximadamente en 180h.

Tabla 3. Partida 1: Mano de obra

Concepto	Participantes	Unidad Básica	Coste unitario(€)	Cantidad(h)	Coste total (€)
Realización del trabajo				300	12000
	Graduado en Ingeniería en Tecnologías Industriales	h	40	1	40
Costes indirectos				2%	240
TOTAL					12240 €

1.4.2. Partida 2: Materiales

Tabla 4. Partida 2: Materiales

Concepto	Unidad Básica	Coste unitario(€)	Cantidad(h)	Coste total(€)
Robot UR3	h	0.57	120	68.4
Ordenador personal(MacBook Pro)	h	0.17	300	51
Torre ordenador laboratorio	h	0.13	20	2.6
Monitor ordenador laboratorio	h	0.037	20	0.74
Placa Arduino Uno	h	N/A	N/A	15
Placa Arduino Ethernet	h	N/A	N/A	25
Módulo reconocimiento voz	h	N/A	N/A	17.5
Cable USB	h	N/A	N/A	2
Cable RJ45	h	N/A	N/A	3
Switch de red de 8 puertos	h	N/A	N/A	30
Microsoft Office	h	0.038	180	6.84
Reprografía				50
SUBTOTAL				272.1
Costes indirectos			2%	5.44
TOTAL				277.54 €

1.5. PRESUPUESTO DE EJECUCIÓN MATERIAL (PEM)

Es la suma de los presupuestos parciales de cada unidad de obra.

Tabla 5. Presupuesto de ejecución material (PEM).

Presupuestos parciales	Coste (€)
Partida 1:Mano de obra	12240
Partida 2: Materiales	277.54
PEM	12517.54 €

1.6. PRESUPUESTO DE EJECUCIÓN POR CONTRATA (PEC)

Para el cálculo de este valor se tienen en cuenta un 13% de gastos generales y un 6% de beneficio industrial

Tabla 6. Presupuesto de ejecución por contrata (PEC).

Presupuesto	Coste (€)
PEM	12517.54
13% Gastos generales	1627.3
6% Beneficio industrial	751.05
PEC	14895.9 €

1.7. PRESUPUESTO BASE DE LICITACIÓN

Finalmente, se tiene en cuenta un 21% de IVA para poder obtenerse el presupuesto base de licitación, que es el valor final del presupuesto que supone este proyecto.

Tabla 7. Presupuesto base de licitación.

Presupuesto	Coste (€)
PEC	14895.9
21% IVA	3128.14
TOTAL	**18024.04 €**

El coste total asciende a DIECIOCHO MIL VEINTICUATRO CON CUATRO CÉNTIMOS.