



**UNIVERSIDAD TÉCNICA DE AMBATO**

**FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E  
INDUSTRIAL**

**CARRERA DE INGENIERÍA EN ELECTRÓNICA Y COMUNICACIONES**

**Tema:**

---

**AUTOMATIZACIÓN DE BAJO COSTO Y CONTROL ROBUSTO PARA  
BRAZOS ROBÓTICOS**

---

Trabajo de Titulación Modalidad: Artículo Académico, presentado previo a la  
obtención del título de Ingeniera en Electrónica y Comunicaciones

**ÁREA: Física y Electrónica**

**LÍNEA DE INVESTIGACIÓN: Sistemas Electrónicos**

**AUTOR: Dalia Solandy Alvarez Montenegro**

**TUTOR: Ing. Geovanni Danilo Brito Moncayo**

**Ambato – Ecuador**

**Agosto – 2020**

## APROBACIÓN DEL TUTOR

En calidad de tutor del Trabajo de Titulación con el tema: AUTOMATIZACIÓN DE BAJO COSTO Y CONTROL ROBUSTO PARA BRAZOS ROBÓTICOS, desarrollado bajo la modalidad Artículo Académico por la señorita Dalia Solandy Alvarez Montenegro, estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 15 del Reglamento para obtener el Título de Tercer Nivel, de Grado de la Universidad Técnica de Ambato, y el numeral 7.4 del respectivo instructivo.

Ambato, agosto 2020.



Firmado electrónicamente por:  
**GEOVANNI DANILO  
BRITO MONCAYO**

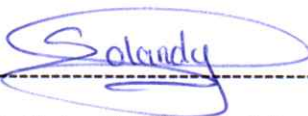
-----  
Ing. Giovanni Danilo Brito Moncayo.

TUTOR

## AUTORÍA

El presente Proyecto de Investigación titulado: AUTOMATIZACIÓN DE BAJO COSTO Y CONTROL ROBUSTO PARA BRAZOS ROBÓTICOS es absolutamente original, auténtico y personal. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, agosto 2020.



Dalia Solandy Alvarez Montenegro  
C.C. 1804314829  
AUTOR

## APROBACIÓN TRIBUNAL DE GRADO

En calidad de par calificador del Informe Final del Trabajo de Titulación presentado por la señorita Dalia Solandy Alvarez Montenegro, estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Artículo Académico, titulado AUTOMATIZACIÓN DE BAJO COSTO Y CONTROL ROBUSTO PARA BRAZOS ROBÓTICOS, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 17 del Reglamento para obtener el Título de Tercer Nivel, de Grado de la Universidad Técnica de Ambato, y al numeral 7.6 del respectivo instructivo. Para cuya constancia suscribimos, conjuntamente con la señora Presidenta del Tribunal.

Ambato, agosto 2020.



ELSA PILAR  
URRUTIA

---

Ing. Pilar Urrutia, Mg.

PRESIDENTA DEL TRIBUNAL



MARCELO  
VLADIMIR GARCIA  
SANCHEZ

---

Dr. Marcelo García  
PROFESOR CALIFICADOR



SANTIAGO MAURICIO  
ALTAMIRANO  
MELENDEZ

---

Ing. Santiago Altamirano  
PROFESOR CALIFICADOR

## DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación.

Cedo los derechos de mi Trabajo de Titulación en favor de la Universidad Técnica de Ambato, con fines de difusión pública. Además, autorizo su reproducción total o parcial dentro de las regulaciones de la institución.

Ambato, agosto 2020.



Dalia Solandy Alvarez Montenegro

C.C. 1804314829

AUTOR

## **DEDICATORIA:**

*Llena de Satisfacción y con todo mi Amor este trabajo se lo dedico a Mis Padres por su apoyo incondicional para el cumplimiento de mis metas propuestas.*

*A mi Madre, mi motivación e inspiración, motor fundamental en mi vida quien con su infinito amor y ardua dedicación ha guiado mis pasos, me ha alentado a ser una mejor persona y a terminar mi carrera universitaria.*

*A mis Hermanos que me han acompañado durante todo este proceso han sido mis cómplices y me han demostrado que con esfuerzo y dedicación se puede lograr alcanzar cualquier ideal propuesto.*

*Dalia Solandy Alvarez Montenegro*

## **AGRADECIMIENTO:**

*A Dios por brindarme la oportunidad de vivir bellas experiencias y alcanzar una meta más en mi vida.*

*A mi querida Institución la Universidad Técnica de Ambato en especial a la Facultad de Ingeniería en Sistemas, Electrónica e Industrial y a la carrera de Ingeniería en Electrónica y Comunicaciones, que me han permitido adquirir todo el conocimiento necesario para mi vida profesional.*

*A los Docentes de mi noble Institución. Al Ing. Geovanni Brito por guiarme y compartir sus conocimientos para poder lograr la culminación de mi artículo científico. Al Dr. Marcelo García, por su confianza y por permitirme participar en el desarrollo del proyecto.*

*A mis Padres y hermanos por su apoyo, consejo y motivación para lograr cumplir mis sueños, sin ellos esta meta cumplida no hubiera sido posible.*

*A todos quienes formaron parte de este proceso de aprendizaje en mi vida universitaria, con quienes he compartido increíbles momentos, mis amigos.*

*Dalia Solandy Alvarez Montenegro*

## ÍNDICE DE CONTENIDOS

<b>A. PÁGINAS PRELIMINARS.....</b>	<b>ii</b>
<b>APROBACIÓN DEL TUTOR .....</b>	<b>ii</b>
<b>AUTORÍA .....</b>	<b>iii</b>
<b>DERECHOS DE AUTOR .....</b>	<b>v</b>
<b>APROBACIÓN DE LA COMISIÓN CALIFICADORA.....</b>	<b>iv</b>
<b>DEDICATORIA: .....</b>	<b>vi</b>
<b>AGRADECIMIENTO:.....</b>	<b>vii</b>
<b>RESUMEN EJECUTIVO .....</b>	<b>ix</b>
<b>ABSTRACT.....</b>	<b>x</b>
<b>CAPÍTULO I.....</b>	<b>1</b>
<b>MARCO TEÓRICO .....</b>	<b>1</b>
<b>1.2    Objetivos .....</b>	<b>4</b>
<b>1.2.1    Objetivo General .....</b>	<b>4</b>
<b>1.2.2    Objetivos Específicos.....</b>	<b>4</b>
<b>CAPÍTULO II .....</b>	<b>5</b>
<b>ARTÍCULO ACEPTADO PARA PUBLICACIÓN .....</b>	<b>5</b>
<b>CAPÍTULO III.....</b>	<b>22</b>
<b>CONCLUSIONES Y RECOMENDACIONES.....</b>	<b>22</b>
<b>CAPITULO IV .....</b>	<b>24</b>
<b>REFERENCIAS .....</b>	<b>24</b>
<b>ANEXOS.....</b>	<b>25</b>



## RESUMEN EJECUTIVO

En los últimos años, se han desarrollado aplicaciones de robótica basadas en código abierto, donde la precisión en los movimientos se considera el objetivo principal al seguir una trayectoria, sin embargo, debido a diferentes circunstancias, pequeños errores pueden interferir con el control individual de las articulaciones del manipulador, Para mejorar la precisión del efector final en el espacio cartesiano, existen diferentes tipos de métodos como la compensación de gravedad que se incluye en el modelo dinámico. Además del desarrollo de un sistema que integra conceptos de la Industria 4.0, se utiliza un controlador de bajo costo para este sistema, donde las altas capacidades de comunicación y procesamiento permiten un rápido intercambio de información entre los componentes. En el presente artículo se desarrolla una aplicación sobre una tarjeta empotrada SBC llamada Raspberry Pi 3B para controlar el brazo del KUKA youBot basado en open source, mediante la compensación de la gravedad se mejora la precisión del efector final al establecer un nuevo nodo ROS que es el principal responsable de la calibración y el control del brazo robótico, finalmente, para la interfaz gráfica, se muestra una GUI desarrollada bajo C++ para el control interactivo del brazo robótico, esta se presenta como un nodo ROS, donde las posiciones se pueden guardar y reproducir, es decir que consta de botones discretos para la ejecución de movimientos mediante el envío de valores posicionales o por variación de velocidad, así como la activación del módulo de control para compensación gravitacional, donde el usuario realiza manualmente las tareas.

**Palabras Clave:** Compensación de la gravedad · Manipuladores de brazos · Control PD· Control Posición · Automatización de bajo costo

## ABSTRACT

In recent years, open source robotics applications have been developed, where precision in movements is considered the main objective when following a trajectory, however, due to different circumstances, small errors can interfere with the individual control of Manipulator joints, To improve the accuracy of the final effector in the Cartesian space, there are different types of methods such as gravity compensation that is included in the dynamic model. In addition to the development of a system that integrates concepts of Industry 4.0, a low-cost controller is used for this system, where high communication and processing capabilities allow rapid exchange of information between components. This article develops an application on an SBC embedded card called Raspberry Pi 3B to control the arm of the KUKA youBot based on open source, by means of gravity compensation the accuracy of the final effector is improved by establishing a new ROS node that will be The main responsible for the calibration and control of the robotic arm, finally, for the graphical interface, a GUI developed under C++ for the interactive control of the robotic arm is shown, this is presented as a ROS node, where the positions can be saved and reproduce, that is to say it consists of discrete buttons for the execution of movements by sending positional values or by speed variation, as well as the activation of the control module for gravitational compensation, where the user manually performs the tasks.

**Keywords:** Gravity compensation • Arm manipulators • PD control • Position control  
• Low cost automation

# CAPÍTULO I

## MARCO TEÓRICO

### 1.1 Antecedentes Investigativos

Del proceso de investigación realizado en los repositorios de diferentes universidades y artículos científicos publicados en revistas, se han encontrado los siguientes proyectos de investigación sobre el tema propuesto.

En el año 2014 la revista Iberoamericana de Automática e Informática Industrial RIAI publica un artículo de Valera Ángel, Vallés Marina y Amparo Soriano, con el tema: **“Plataformas de Bajo Coste para la Realización de Trabajos Prácticos de Mecatrónica y Robótica”**. Enuncia que: hoy en día las tarjetas microcontroladoras gracias a su versatilidad y bajo costo han llegado a tener peso en la robótica educacional y de investigación. A demás se indica que el objetivo del micro ordenador de nombre Raspberry PI es promover el uso del mismo en el campo académico, donde el aspecto más destacable es su utilización en el campo de la robótica [1].

En el año 2015 la revista XXXVI Jornadas de Automática en Bilbao publica un artículo de González Pablo, Calvo Isidro, Etxeberria Ismael, Zulueta Ekaitz, López José, con el tema: **“Hacia un framework basado en ROS para la implementación de Sistemas Ciber-físicos”**. Enuncia que: El software ROS es un middleware creado para simplificar la comunicación y la escritura para el control de robots, el cual se instala sobre GNU/Linux permitiendo incorporar varios dispositivos a una red modular y distribuida, donde un nodo maestro permite al resto de nodos comunicarse por TCP/IP, además dichos elementos que forman parte de la arquitectura de CPS se pueden asegurar mediante la implementación de un servidor VPN, se obtienen varias ventajas de esta red de nodos ROS como la modularidad de la arquitectura permitiendo la adaptación a otros sistemas, es posible la redundancia debido al esquema publicación/suscriptor de ROS, además de permitir el acceso multiplataforma[2].

En el año 2015 la revista International Journal of Innovative and Emerging Research in Engineering, publica un artículo de Shri Sant Gadge Baba, con el tema: **“Raspberry Pi Technology: A Review”**. Enuncia que: El desarrollo de proyectos mediante la utilización de Raspberry Pi basados en un procesador ARM con la ejecución de Linux o cualquiera de sus distribuciones disponibles se asienta como el cerebro perfecto para varios robots mediante aplicaciones realizadas en ROS, además mediante el enfoque de código abierto Linux ha llegado a ser la plataforma más utilizada en la robótica[3].

En el año 2015 la revista INGENIARE, Universidad Libre-Barranquilla, publica un artículo de Marvin Molina, Patty Pedroza, Kevin Gaitán, Javier Salgado, María Ordóñez con el tema: **“Design and construction of a Robotic Arm Prototype with three degrees of freedom as an object of study”**. Enuncia que: la robótica ha permitido la creación de dispositivos que ayudan a automatizar las tareas del ser humano, un ejemplo de esto son los brazos robóticos, los cuales se han convertido en una herramienta para profesionales y estudiantes, debido a que permite realizar tareas complejas, peligrosas y repetitivas de manera más sencilla y eficiente[4].

En el año 2015 la revista IV CONGRESO INTERNACIONAL DE INGENIERÍA MECATRÓNICA Y AUTOMATIZACIÓN, Universidad Libre-Barranquilla, publica un artículo de Kewin J. Saumeth, Federico Pinilla, A. Fernández Arboleda, David J. Muñoz, Luis A. Cruz con el tema: **“Sistema Ciber-Físico de una CNC para la producción de circuitos impresos”**. Enuncia que: Uno de los desarrollos más significativos de los sistemas electrónicos y su integración con la tecnología de la información y las tendencias de comunicación, está representado por Cyber Physical Systems (CPS), los cuales están asociados con los sistemas informáticos, que están estrechamente relacionados con: el mundo físico y sus procesos; proporcionar y usar al mismo tiempo, servicios de acceso a datos y procesamiento de información, ambos disponibles en Internet; reconfigurabilidad de sistemas y procesos de control; y en general con las características de Internet de las cosas (IoT), incluido el tipo industrial o IoT[5].

En junio del 2016, en la Universitat Oberta de Catalunya, Javier Gutiérrez Pérez presenta una tesis con el tema: **“Introducción a ROS en Raspberry Pi”**. Se menciona que: Al momento de elegir plataformas compatibles con ROS se debe considerar las limitaciones en la comunicación, además que al momento de desarrollar software para robots existe una gran variedad de lenguajes de código abierto como C++, Python o Java. También se enuncia que el sistema empujado Raspberry Pi es una de las mejores elecciones debido a su capacidad de soportar ROS además de su bajo costo, sin embargo, se debe realizar una minuciosa selección debido a que no todos los softwares montables en la tarjeta son compatibles y poseen los repositorios necesarios[6].

En el año 2018 la revista Enfoque UTE publica un artículo de García Marcelo, Irisarri Edurne, Pérez Federico, con el tema: **“Integración Vertical en plantas industriales utilizando OPC UA e IEC-61499”**. Enuncia que: Los Sistemas Ciber-físicos de Producción (CPPS–Cyber-Physical Production Systems) CPPSs permiten la integración de sistemas de adquisición de datos tradicionales y novedosos sistemas de procesamiento inteligente de datos, con el objetivo de extraer información y mejorar el rendimiento general del sistema productivo. Para lograrlo, es necesario cerrar la brecha existente entre los sistemas de control y los niveles superiores, para ello proponen una aproximación en el desarrollo de aplicaciones bajo la norma IEC-61499 para el intercambio de datos entre el nivel de planta y las capas más altas empleando el estándar industrial OPC UA[7].

En el año 2018 la revista Iberoamericana de Automática e Informática industrial publica un artículo de García Marcelo, Irisarri Edurne, Pérez Federico, Estévez Elisabet, Marcos Marga con el tema: **“Arquitectura de Automatización basada en Sistemas Ciber-físicos para la Fabricación Flexible en la Industria de Petróleo y Gas”**. Enuncia que: la utilización de Sistemas de Producción Ciber-Físicos se basa en dispositivos de control con una amplia capacidad de comunicación que puede ser local o remota, además de forma colectiva con el Internet Industrial de las Cosas proveen de tecnologías de impacto en todos los aspectos de las empresas manufactureras haciendo necesaria la inducción de avances industriales que promuevan el desarrollo basado en modelos para sistemas de control distribuidos[8].

## **1.2 Objetivos**

### **1.2.1 Objetivo General**

- Desarrollar la automatización sobre plataformas de bajo costo y control robusto para brazos robóticos.

### **1.2.2 Objetivos Específicos**

- Establecer el entorno de simulación para el control del brazo robótico del manipulador.
- Analizar los diferentes parámetros de comunicación para el control de las juntas rotatorias del brazo robótico.
- Diseñar el sistema control con una interfaz gráfica para el posicionamiento y velocidad del manipulador Kuka youBot.

## CAPÍTULO II

### ARTÍCULO ACEPTADO PARA PUBLICACIÓN

El presente artículo fue elaborado en Idioma Inglés y aceptado para la publicación en la revista *Advances in Intelligent Systems and Computing Series* Ed.: Kacprzyk, Janusz. Originalmente publicado con el título: *Advances in Intelligent and Soft Computing* ISSN: 2194-5357, su traducción en el idioma español se detalla a continuación.

#### Resumen

En los últimos años, se han desarrollado aplicaciones de robótica basadas en código abierto, donde la precisión en los movimientos se considera el objetivo principal al seguir una trayectoria, sin embargo, debido a diferentes circunstancias, pequeños errores pueden interferir con el control individual de las articulaciones del manipulador, Para mejorar la precisión del efector final en el espacio cartesiano, existen diferentes tipos de métodos como la compensación de gravedad que se incluye en el modelo dinámico. Además del desarrollo de un sistema que integra conceptos de la Industria 4.0, se utiliza un controlador de bajo costo para este sistema, donde las altas capacidades de comunicación y procesamiento permiten un rápido intercambio de información entre los componentes. En este artículo se desarrolla una aplicación sobre Raspberry Pi 3B para controlar el brazo de KUKA youBot a través de una interfaz gráfica donde las posiciones se pueden guardar y reproducir.

**Palabras Clave:** Compensación de la gravedad · Manipuladores de brazos · Control PD· Control Posición · Automatización de bajo costo

#### 2.1 Introducción

El control de procesos a través de sistemas integrados está pasando por una etapa de crecimiento gracias a la apertura de industrias a sistemas cuyas capacidades de monitoreo y control han mejorado su modo de operación a pesar de su bajo costo, el

progreso de las aplicaciones basadas en Internet permite entornos de producción distribuidos a través de la interacción entre los sistemas informáticos integrados y los entornos físicos, dando lugar a procesos físicos automáticos y al control de los sistemas físicos cibernéticos (CPS), lo que demuestra el alto potencial que tiene en el sector productivo, además, debe tenerse en cuenta que un sistema completo debe tener varios requisitos, que tienen diferentes aplicaciones con sus propias barreras [9].

El uso del Sistema Operativo Robot (ROS) como un marco implementable dentro de los Sistemas Ciberfísicos, ya que simplifica la comunicación y el monitoreo del estado de control del robot, este llamado software está instalado en GNU / Linux permitiendo la incorporación de varios dispositivos. en una red distribuida, donde un nodo maestro permite que el resto de los nodos se comuniquen a través de TCP / IP, además, dichos elementos que forman parte de la arquitectura CPS se pueden asegurar mediante la implementación de un servidor VPN, donde varias ventajas de esta red de los nodos ROS que se obtienen, como la modularidad de la arquitectura que permite la adaptación a otros sistemas, es posible la redundancia debido al esquema de publicación / suscriptor ROS, además de permitir el acceso multiplataforma y el control remoto a través de otro elemento conectado a la misma red [4].

Para el control de las articulaciones del brazo robótico se han definido dos métodos, un cambio de la posición actual a la siguiente o una variación en la velocidad de una articulación hasta alcanzar la posición deseada. Para ambos, se requiere una calibración adicional considerando la compensación gravitacional, lo que permite el monitoreo de las trayectorias del efector final de una manera apropiada, reduciendo el error en la posición y la velocidad [10].

El objetivo de esta investigación es presentar el diseño de un sistema de control avanzado a través de la compensación gravitacional utilizando una plataforma de hardware de bajo costo del tipo SBC que permite la programación bajo la interfaz de programación de aplicaciones (API) del brazo robótico Kuka youBot, que incluye métodos y clases para el desarrollo de programas, además del uso de ROS, que permiten el control sobre controladores de código abierto, lo que lleva a la interoperabilidad entre sistemas independientes.



El contenido del artículo está estructurado de la siguiente manera: la Sección 2 proporciona conceptos breves que permite una mejor comprensión de las siguientes secciones. La Sección 3 presenta el estudio de caso utilizado para el método de investigación. Mientras que en la Sección 4, se muestra el diseño e implementación de una compensación por gravedad para el control del robot basada en la automatización de bajo costo. La discusión de los resultados se presenta en la Sección 5. Finalmente, las conclusiones y el trabajo futuro se presentan en la Sección 6.

## **2.2 Estado de la Tecnología**

### **2.2.1 KUKA youBot**

Se define como un manipulador móvil desarrollado principalmente para educación e investigación, que consta de dos partes que se pueden usar de forma independiente, un brazo robótico de cinco grados de libertad con dos dedos como abrazadera y una base omnidireccional, la API del brazo es compatible con ROS y puede comunicarse a través de Ethernet y EtherCAT, y en su base tiene una mini PC ITX integrada con su CPU [11].

Como se muestra en la Figura. 1, el brazo robótico consta de cinco articulaciones giratorias y una abrazadera de dos dedos en paralelo como herramienta final, alcanza una altura de 655 mm, fue diseñado para transportar una carga de hasta 0,5 kg y sus articulaciones rotativas Se puede acceder a través de Ethernet y EtherCAT, las juntas dos, tres y cuatro tienen su movimiento de rotación paralelo a un eje, mientras que las juntas uno y cinco giran en paralelo a otra cada vez que el brazo apunta hacia arriba, la velocidad máxima de rotación de cada junta es de 90 grados/s mientras que el efector final se puede abrir 11.5 mm para cada dedo [3].



**Fig. 1:** Brazo del Kuka youBot

**Elaborado por:** Investigadora

El protocolo de comunicación definido para el Kuka YouBot utiliza la red Ethernet estándar del tipo Maestro de código abierto simple EtherCAT (SOEM), que permite la implementación de un controlador de código abierto para el maestro [3].

El controlador youBot tiene una biblioteca con clases definidas en C++ que permiten el control y monitoreo de actuadores y sensores, proporciona las funcionalidades para establecer valores de posición y velocidad cartesiana, la API proporcionada por el controlador contiene subsistemas para manejar las articulaciones donde hay clases para el brazo y la base [2].

### **2.2.2 Compensación por gravedad**

El campo de control de estabilidad de los robots manipuladores se ha vuelto más importante desde su propuesta en 1981, ya que esto solo se aplica a los brazos robóticos que solo tienen uniones giratorias, para las ecuaciones de modelado matemático que se usan no lineales, sin embargo, la teoría de Lyapunov se usa a menudo como un medios auxiliares [8].

El desarrollo del control PD con compensación de gravedad para manipuladores se basa en el supuesto de que la dinámica de los actuadores no existe, por lo que se debe considerar que la señal de control no es el par, sino el voltaje que se aplica a los motores, donde el par viene como resultado de la interacción eléctrica de los actuadores [1].

Sin considerar las perturbaciones externas, se describe el modelo matemático de un robot rígido que tiene un número  $n$  de eslabones (ver Ecuación. 1):

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + F(\dot{q}) = \tau \quad (1)$$

Donde  $M(q)$  representa la matriz de inercia simétrica del manipulador,  $C(q, \dot{q})\dot{q}$  es la matriz de fuerzas centrípetas y coriolis,  $G(q)$  es el vector de pares gravitacionales,  $F(\dot{q})$  es pares de fricción y  $\tau$  representa el vector de torque de entrada [1].

El algoritmo de control proporcional derivativo con compensación de gravedad se define mediante la corrección del error de posicionamiento de la articulación (ver Ecuación.2):

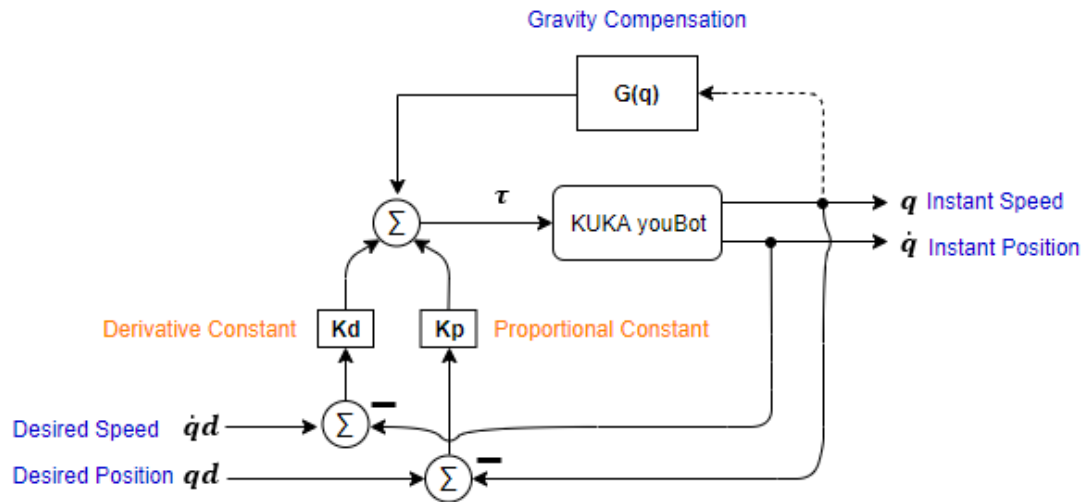
$$\tau = k_p \tilde{q} - k_d(\dot{\tilde{q}}) + G(q) \quad (2)$$

Donde  $k_p \tilde{q}$  es la ganancia proporcional multiplicada por el error de posición,  $k_d(\dot{\tilde{q}})$  es la ganancia derivativa multiplicada por la velocidad de movimiento y  $G(q)$  representa la compensación del par gravitacional.

La compensación gravitacional se define como un término proporcional al error de posición, al que se agrega un amortiguamiento que consiste en la velocidad de movimiento, esta parte del control consiste en una retención mecánica que absorbe los impulsos resultantes de la respuesta, suprimiendo picos y oscilaciones, Para la etapa estacionaria, se estima que la posición del efector o herramienta terminal es constante, suponiendo un valor de cero a la velocidad de movimiento [5].

De acuerdo con lo expresado en [6], el bucle de control con compensación de gravedad para robots se establece como una ecuación que necesita una medición constante de la

posición y la velocidad, la ecuación (2) determina el modelo dinámico, el modelo dinámico para el control de DP con gravitacional compensación por robots manipuladores.



**Fig. 2.** Modelo Dinámico para control PD con compensación de gravedad

**Elaborado por:** Investigadora

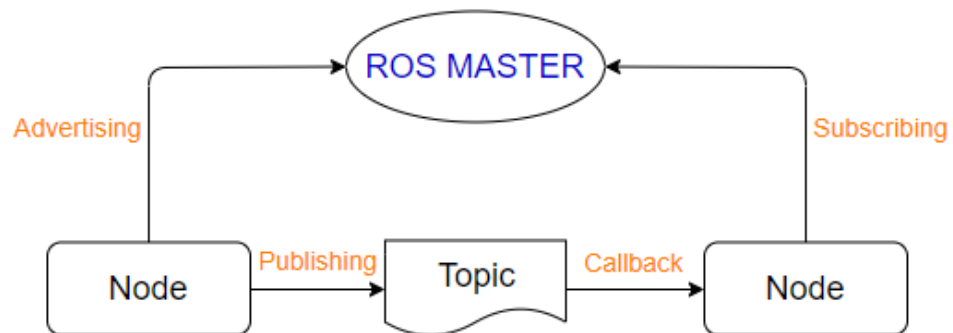
Además, la figura 2 muestra el diagrama de bloques equivalente a la ley de control PD con compensación de gravedad, donde  $K_p$  y  $K_d$  se definen como matrices simétricas positivas, sin embargo, este tipo de optimización requiere un conocimiento parcial del manipulador, considerando los valores de longitud, masa y parámetros rotacionales de los componentes del brazo robótico.

### 2.2.3 ROS Sistema Operativo Robot

Es un espacio de trabajo de código abierto que facilita y acelera la construcción de aplicaciones robóticas como la entrega de mensajes, la gestión de paquetes y los controladores de dispositivos de hardware. Originalmente diseñado para simplificar los sistemas distribuidos, como se ve en la Figura. 3, los bloques de construcción de una aplicación ROS se definen como nodos, donde cada uno es una entidad que puede comunicarse con otros nodos que son parte de un proceso del sistema operativo, los nodos soportados por ROS están escritos principalmente en C++ y Python. Los nodos en un entorno ROS pueden comunicarse de tres maneras, tales como: publicar

mensajes en un tema, escuchar mensajes publicados en un tema y llamar a un servicio proporcionado por otro nodo [7].

Inicialmente para la gestión del sistema de comunicación, se debe iniciar un ROS maestro, que es responsable de permitir que los nodos se reúnan e intercambien mensajes, el uso de temas o temas se define como rutas de conexión lógica, donde los nodos pueden publicar temas o suscribirse a ellos

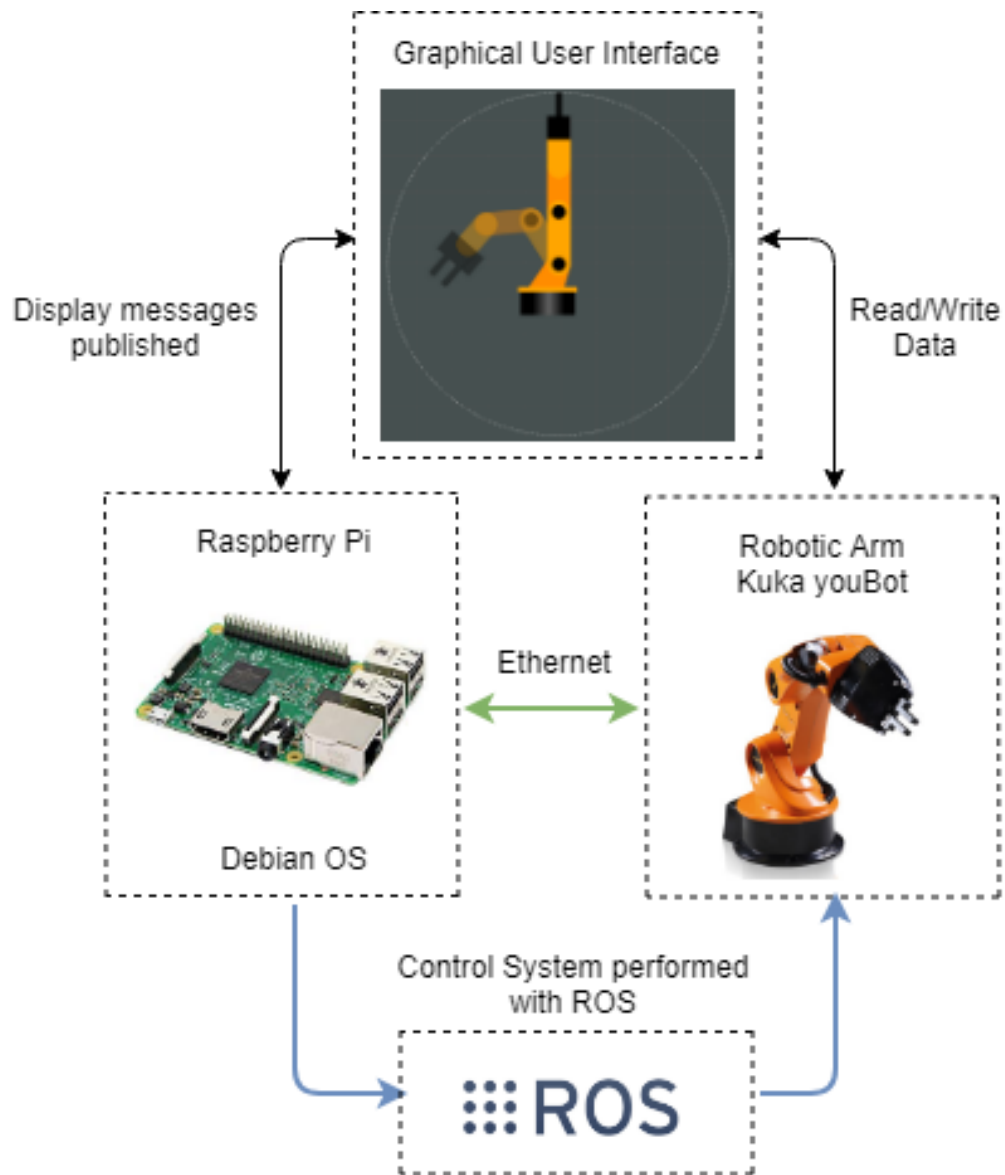


**Fig. 3.** Component system of a ROS model

**Elaborado por:** Investigadora

### 2.3 Caso de Estudio

El posicionamiento de objetos es una de las tareas más comunes que realizan los brazos robóticos, este tipo de procedimiento requiere un control que permita manipular las articulaciones rotativas de acuerdo con la necesidad del proceso.



**Fig. 4:** Arquitectura del caso de estudio

**Elaborado por:** Investigadora

El presente trabajo muestra el diseño de un sistema de control para el brazo robótico del manipulador Kuka youBot desarrollado en la tarjeta integrada Raspberry Pi 3B, esto para la realización de una tarea de posicionamiento de "escoger y colocar" y el monitoreo a través de la tabla de interfaz que tiene La opción de guardar y reproducir posiciones, como se ve en la Fig. 4, la arquitectura se genera a través de la integración y el trabajo multiplataforma de ROS y el controlador youBot, lo que permite el desarrollo de bibliotecas, temas y nodos, en los que los datos de comunicación y los mensajes serán enviados y recibidos a través del protocolo Ethernet.

La interfaz gráfica permite el control mediante el envío de valores de posición y velocidad a través del controlador de brazo robótico, para lo cual se desarrollan clases y métodos que permiten enviar datos a las juntas rotativas en C ++, además de la manipulación a través del control de compensación gravitacional, mediante el cual la entrega controlada de corriente a los motores de la junta asegura que no haya daños en las juntas del manipulador.

En la presente investigación se utilizan tarjetas integradas de tipo SBC de bajo costo porque se consideran controladores o posibles soluciones a los problemas de procesamiento y monitoreo de datos, siendo compatibles con código abierto, el control está diseñado con compensación gravitacional, que consiste en la interpolación de posiciones enviadas, de modo que los valores actuales enviados a los motores están regulados proporcionalmente y de forma derivada, para que la enseñanza de las posiciones sea más versátil, se incluye este método de control, esto permite que un operador maneje el brazo robótico de acuerdo con sus necesidades manualmente.

## **2.4 Propuesta de Implementación**

### **2.4.1 Compensación Gravitatoria**

Se establece un nuevo nodo ROS que será el principal responsable de la calibración y el control del brazo robótico, para esto se han configurado previamente los parámetros físicos requeridos por el módulo de compensación gravitacional, como fricción, masa, gravedad, etc. Han sido previamente configurados.

Cuando se inicia el módulo de compensación gravitacional, se cargan los parámetros mencionados anteriormente, de forma simultánea y automática se realiza la calibración inicializada del brazo robótico, cada junta rotativa se define como un esclavo EtherCAT, y para cada una de ellas se realizan iteraciones para encontrar el valor más exacto en el que se reduce el par producido por los motores, esto se debe a que los cambios de par son proporcionales a los de la corriente.

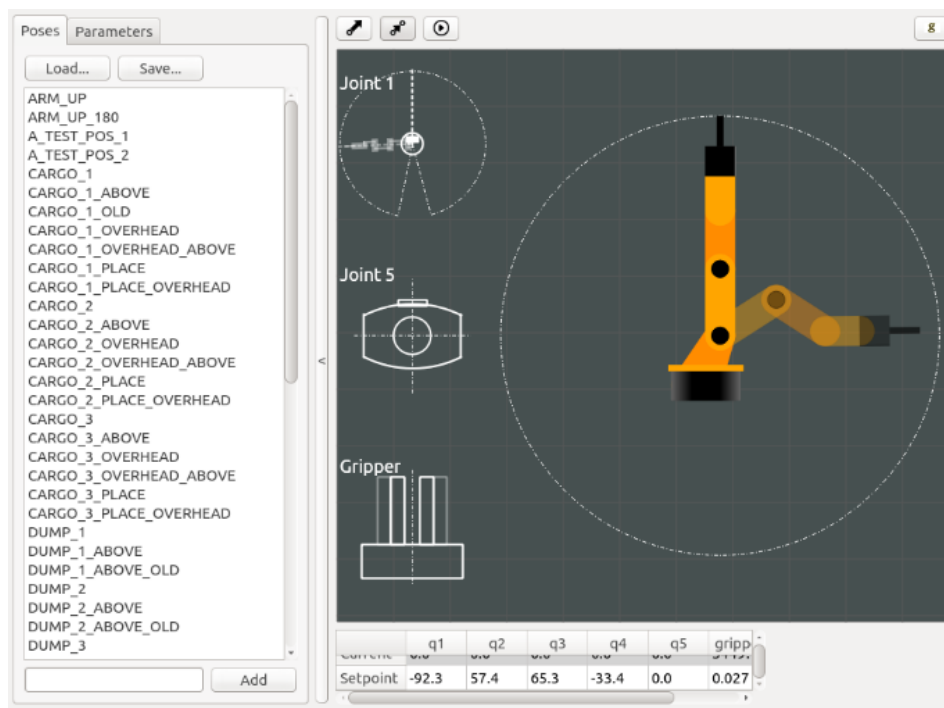
Principalmente basado en el uso de la biblioteca nlopt, desarrollada para sistemas de optimización no lineal, esto permite la evaluación de varios algoritmos para rutinas de calibración largas, lo que permite probar varios parámetros en un corto período de tiempo.

Cuando el módulo se activa desde la GUI, los valores actuales entregados a las uniones son los mínimos para que pueda mantener los enlaces estáticamente, por lo tanto, cuando se aplica una fuerza externa para manipular el brazo robótico, no presenta oposición, ya que que la operación del control se verifica mediante compensación gravitacional.

#### **2.4.2 Interfaz Gráfica de Usuario**

La GUI desarrollada para el control interactivo del brazo robótico se presenta como un nodo ROS, que intercambia datos con el nodo controlador y este a su vez es responsable del control del dispositivo de hardware a través de un tercer nodo implementado con el robot en formato URDF, la interfaz se desarrolla bajo C++ y consta de botones discretos para la ejecución de movimientos mediante el envío de valores posicionales o por variación de velocidad, así como la activación del módulo de control para compensación gravitacional, también tiene la función de guardar y la carga de estados, para que el usuario pueda realizar manualmente las tareas de posicionamiento planteadas en la investigación





**Fig. 5:** Interfaz gráfica de Usuario

**Elaborado por:** Investigadora

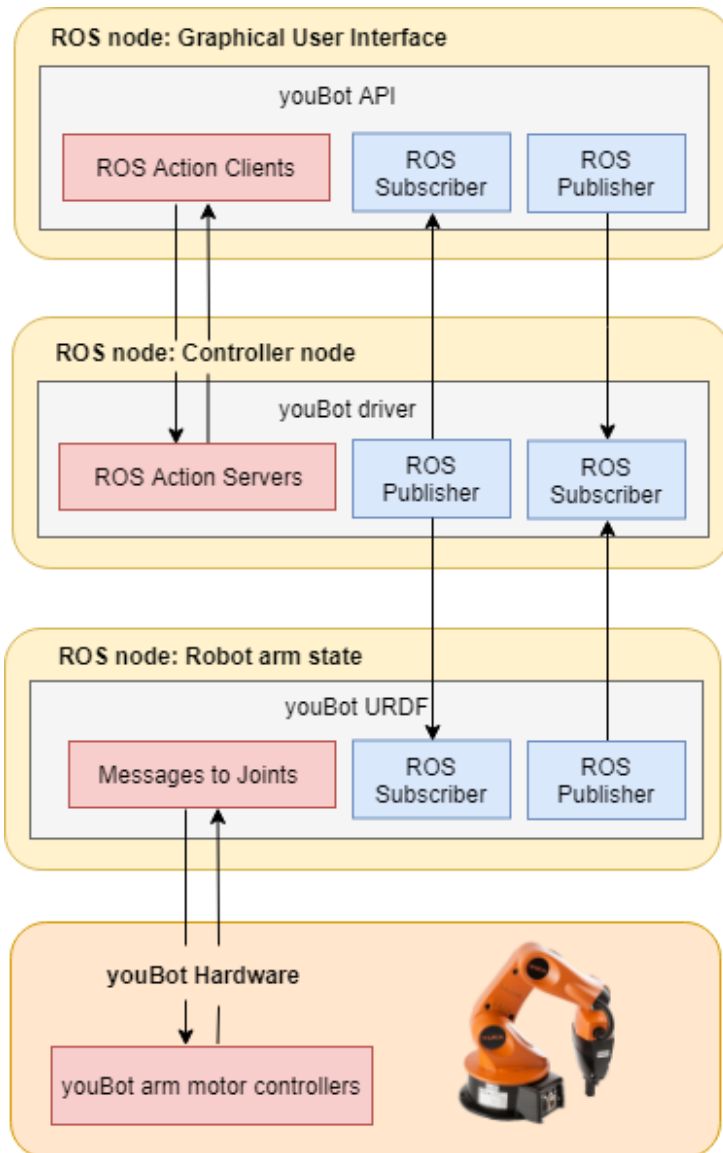
Dentro de la GUI, además de la paleta para cargar y guardar estados de brazo, como se muestra en la Figura. 5, una visualización 2D del brazo robótico, donde se presentan dos etapas de esta, una actual y otra que comprenderá el objetivo para el cual el brazo se debe mover, para comprender el posible movimiento realizado en el plano faltante, se incluyen tres vistas que corresponden al detalle superior de la primera junta, la junta número 5 para la rotación de la pinza y un frente de esta última para visualizar correctamente la apertura o cierre de la misma. En la parte inferior, se observan los parámetros de rotación actuales de las cinco articulaciones más la variación longitudinal de la pinza, para mantener los valores posicionales dentro de los rangos permitidos de cada motor.

### 2.4.3 ROS System

Para la implementación del caso de estudio propuesto inicialmente, se desarrolla un sistema ROS con tres nodos como se ve en la figura 5, que permite el control sobre el brazo robótico del manipulador Kuka youBot a través de un método avanzado basado en la compensación gravitacional, si los tres los nodos son suscriptores y editores de

diferentes temas, esto para que los mensajes enviados desde la GUI desarrollada lleguen a través del nodo controlador a las reuniones físicas rotativas, dentro de la GUI las acciones a ejecutar por cada componente se detallan a través de archivos en C++ de la misma, se describe a través de la API youBot cómo el mensaje es publicado y leído por el nodo del controlador, que es el núcleo del sistema completo, esto inicializa el maestro ROS.

Además de ejecutar el controlador de brazo robótico con sus respectivas configuraciones y principalmente el módulo de compensación gravitacional con sus ciertos parámetros de calibración, el nodo transmite los mensajes enviados desde la GUI a través de th El robot en formato URDF, que permite el acceso en un tercer nodo al dispositivo de hardware a través del protocolo EtherCAT, este último nodo describe la geometría de varios elementos físicos, con los cuales el controlador asegura que el robot inicializado sea el correcto para la API utilizado, traduciendo los mensajes enviados a los motores como valores de posición, velocidad o par en variaciones de corriente o límites específicos.

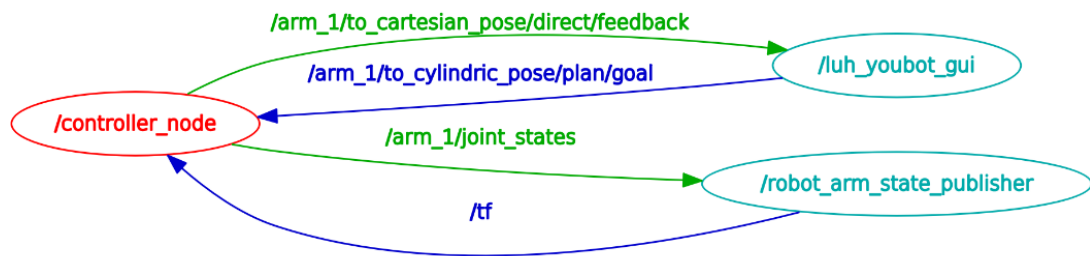


**Fig. 6:** Arquitectura final

**Elaborado por:** Investigadora

La herramienta ROS Rqt Graph se utiliza para obtener un diagrama en tiempo real de la iteración entre nodos, temas y mensajes, donde puede observar el flujo de información dentro del sistema ROS desarrollado, el tipo de mensajes utilizados para proporcionar control sobre las juntas rotativas de el brazo robótico permite controlar los motores a través del controlador youBot, en la Figura. 7, se presenta el gráfico obtenido, que es equivalente al sistema propuesto como caso de estudio.

El tipo de mensajes enviados desde el nodo controlador al nodo del robot URDF, son `sensor_msgs / JointState`, estos se utilizan para transmitir valores en formato `float64` dentro de vectores, para esto se define un vector de cinco posiciones, donde cada uno de ellos estará ocupado por cada una de las juntas rotativas del brazo robótico, esto corresponde a los cinco grados de libertad y lo establece la API de `youBot`, estos espacios se llenarán cuando se envíe un valor de tipo de posición, velocidad o esfuerzo a la matriz, cuando un la posición no se ha llenado, se asume un valor de cero y esto no afecta la ejecución de la orden enviada al brazo robótico.



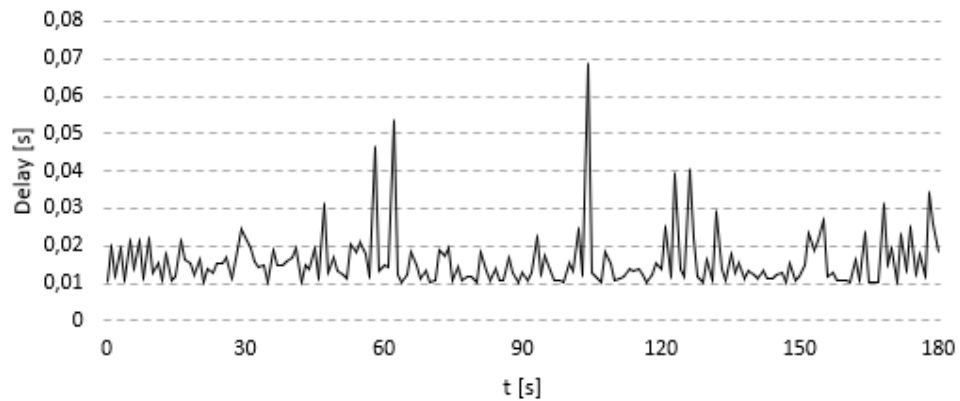
**Fig. 7:** Rqt Graph

**Elaborado por:** Investigadora

## 2.5 Resultados y Discusión

Para verificar el funcionamiento del nodo controlador desarrollado, todos los valores medidos superiores a 0.01 segundos se establecen como un retraso, las mediciones se definen en el archivo ejecutable del nodo mencionado, donde la diferencia entre la frecuencia establecida y la operación del brazo robótico se calcula, el retraso en la operación del controlador en función del tiempo se obtiene del valor inverso del resultado.

El estudio de caso presenta una tarea de posicionamiento llamada "pick and place", para la cual se utilizó el módulo de compensación gravitacional para el control manual del brazo robótico hacia las posiciones deseadas, además, el almacenamiento y la reproducción de las posiciones se utilizaron a través de la GUI para comprobar el funcionamiento de la misma, finalmente se obtuvieron dos puestos para el desarrollo de la tarea.



**Fig. 8:** Retrasos obtenidos en el caso de estudio

**Elaborado por:** Investigadora

Como se puede ver en la figura 8, que es una representación gráfica de los retrasos obtenidos en función del tiempo, el controlador no tiene grandes retrasos en la ejecución del módulo de compensación gravitacional, ya que estos permanecen cerca del límite establecido de 0.01 seg, excepto en casos extremos donde los valores sufren picos relativamente altos.

## 2.6 Conclusiones and Trabajo Futuro

Este trabajo muestra el diseño de un nodo controlador dentro de un sistema desarrollado a través de la API ROS y el controlador youBot, montado en una tarjeta SBC de bajo costo, la arquitectura propuesta como estudio de caso aumenta la interoperabilidad entre plataformas de hardware basadas en código abierto, la elaboración de la aplicación se considera como un metapaquete que contiene los paquetes y archivos necesarios para el control del brazo robótico del manipulador Kuka youBot a través de EtherCAT. Las tareas de calibración y control de las juntas rotativas del brazo robótico para el control a través del módulo de compensación gravitacional, se realizan en segundo plano y automáticamente una vez que se ha inicializado el nodo del controlador, cuando finaliza este proceso, se puede ejecutar la GUI desarrollada, que permite un control más intuitivo para el usuario, ya que permite observar la posición actual y el objetivo al que desea moverse, además de hacer posible la selección del tipo de control, ya sea por posición, velocidad o por la compensación gravitacional módulo.

El controlador funciona constantemente desde que se ejecutó el maestro ROS y todo el sistema, para una operación correcta se debe establecer un valor de 200Hz para la frecuencia del brazo robótico, que varía para cada articulación gracias a los parámetros de operación reales, esto crea un valor existente retraso en el bucle de control, que se muestra en el terminal y permite observar en función del tiempo cuánto se retrasa el seguimiento del proceso de selección y colocación desarrollado, obteniendo finalmente valores bajos que varían de acuerdo con la precisión en las posiciones que deben ser alcanzado por el brazo robótico.

Las futuras líneas de investigación deberían centrarse en el desarrollo de diferentes métodos de control basados en la predicción de trayectorias a través de cinemática directa o inversa, en el uso de otros tipos de software o protocolos que permitan mayores límites en la interoperabilidad del hardware, a través de la aplicación de bibliotecas para el procesamiento de imágenes y el uso de diferentes sensores para la detección de objetos.

## 2.5 Referencias

1. Binazadeh, T., Yousef, M.: Designing a cascade-control structure using fractional-order controllers: Time-delay fractional-order proportional-derivative controller and fractional-order sliding-mode controller. *Journal of Engineering Mechanics* 143(7), 04017037 (2017). [https://doi.org/10.1061/\(ASCE\)EM.1943-7889.0001234](https://doi.org/10.1061/(ASCE)EM.1943-7889.0001234).
2. Di Napoli, G., Filippeschi, A., Tanzini, M., Avizzano, C.A.: A novel control strategy for youbot arm. In: *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*. pp. 482-487 (Oct 2016). <https://doi.org/10.1109/IECON.2016.7793658>.
3. Garcia, C.A., Franklin, S.L., Marino, C., Villalba, W.R., Garcia, M.V.: Design of exible cyber-physical production systems architecture for industrial robot control. In: *2018 IEEE Third Ecuador Technical Chapters Meeting (ETCM)*. pp. 1-6 (Oct 2018). <https://doi.org/10.1109/ETCM.2018.8580338>.
4. Garcia, C.A., Lanas, D., Edison, A.M., Altamirano, S., Garcia, M.V.: An approach of cyber-physical production systems architecture for robot control. In: *IECON*

- 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society. pp.2847{2852 (Oct 2018). <https://doi.org/10.1109/IECON.2018.8591286>.
5. de Gea Fern\_andez, J., Mronga, D., Gunther, M., Knobloch, T.,Wirkus, M., Schroer, M., Trampler, M., Stiene, S., Kirchner, E., Bargsten,V., Banziger, T., Teiwes, J., Kruger, T., Kirchner, F.: Multimodal sensor-based whole-body control for human{robot collaborationin industrial settings. *Robotics and Autonomous Systems* 94, 102 {119 (2017). <https://doi.org/https://doi.org/10.1016/j.robot.2017.04.007>, <http://www.sciencedirect.com/science/article/pii/S0921889016305127>.
  6. Hern\_andez, V., Santib\_a~nez, V., Carrillo, R., Molina, J., L\_opez, J.: Control pd de robots: Din\_amica de actuadores y nueva sinton\_\_a. *Revista Iberoamericana de Autom\_atica e Inform\_atica Industrial RIAI* 5(4), 62 { 68 (2008). [https://doi.org/https://doi.org/10.1016/S1697-7912\(08\)70178-X](https://doi.org/https://doi.org/10.1016/S1697-7912(08)70178-X), <http://www.sciencedirect.com/science/article/pii/S169779120870178X>.
  7. Koub^aa, A.: *Robot Operating System (ROS)*. Springer (2017). <https://doi.org/https://doi.org/10.1007/978-3-319-26054-9>.
  8. Peidr\_o, A., Reinoso, \_ O., Gil, A., Mar\_\_n, J.M., Pay\_a, L.: An\_alisis de estabilidad de singularidades aisladas en robots paralelos mediante desarrollos de taylor de segundo orden (2017).
  9. Wang, L., T •orngren, M., Onori, M.: Current status and advancement of cyber-physical systems in manufacturing. *Journal of Manufacturing Systems* 37, 517 { 527 (2015). <https://doi.org/https://doi.org/10.1016/j.jmsy.2015.04.008>, <http://www.sciencedirect.com/science/article/pii/S0278612515000400>.
  10. Yu, C., Li, Z., Liu, H.: Research on gravity compensation of robot arm based on model learning\*. In: 2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM). pp. 635{641 (July 2019). <https://doi.org/10.1109/AIM.2019.8868673>.
  11. Zhang, B., Gao, S.: Kuka youbot arm path planning based on gravity. In: 2018 International Conference on Mechanical, Electrical, Electronic Engineering Science (MEEES 2018). Atlantis Press (2018/05). <https://doi.org/https://doi.org/10.2991/mees-18.2018.75>, <https://doi.org/10.2991/mees-18.2018.75>.

## CAPÍTULO III

### CONCLUSIONES Y RECOMENDACIONES

#### 3.1 Conclusiones

- El entorno de simulación se estable en Ubuntu, mediante la terminal de ubuntu en windows que se habilita en las opciones de programador y utilizando un subsistema de Linux para Windows (WSL), en la terminal se instala ROS kinetic y las dependencias necesarias, se crea un espacio de trabajo catkin en el cual se clona el repositorio git para realizar la simulación, se compila el espacio de trabajo para la construcción del plugin que permitirá el envío de mensajes hacia el kuka youbot simulado.
- Este trabajo muestra el diseño de un nodo controlador dentro de un sistema desarrollado a través de la API ROS y el controlador youBot, montado en una tarjeta SBC de bajo costo, la arquitectura propuesta como estudio de caso aumenta la interoperabilidad entre plataformas de hardware basadas en código abierto, la elaboración de la aplicación se considera como un metapaquete que contiene los paquetes y archivos necesarios para el control del brazo robótico del manipulador Kuka youBot a través de EtherCAT. Las tareas de calibración y control de las juntas rotativas del brazo robótico para el control a través del módulo de compensación gravitacional, se realizan en segundo plano y automáticamente una vez que se ha inicializado el nodo del controlador, cuando finaliza este proceso, se puede ejecutar la GUI desarrollada, que permite un control más intuitivo para el usuario, ya que permite observar la posición actual y el objetivo al que desea moverse, además de hacer posible la selección del tipo de control, ya sea por posición, velocidad o por la compensación gravitacional módulo.
- El controlador funciona constantemente desde que se ejecutó el maestro ROS y todo el sistema, para una operación correcta se debe establecer un valor de 200Hz para la frecuencia del brazo robótico, que varía para cada articulación gracias a los



parámetros de operación reales, esto crea un valor existente retraso en el bucle de control, que se muestra en el terminal y permite observar en función del tiempo cuánto se retrasa el seguimiento del proceso de selección y colocación desarrollado, obteniendo finalmente valores bajos que varían de acuerdo con la precisión en las posiciones que deben ser alcanzado por el brazo robótico.

### **3.2 Recomendaciones**

- Al usar Linux como subsistema para Windows es necesario usar un servidor Xming para la ejecución de entornos gráficos, se recomienda usar xlaunch que permite configurar el número de pantallas, la manera de iniciar, parámetros y otras configuraciones.
- Se debe tener en cuenta que para mejorar el rendimiento de ROS, este se actualiza cada año en casi todos los repositorios, por lo cual es necesario actualizar ciertos drivers, reescribiendo paquetes disponibles y dependencias actualizadas, para trabajar con versiones de ROS más estables
- Las futuras líneas de investigación deberían centrarse en el desarrollo de diferentes métodos de control basados en la predicción de trayectorias a través de cinemática directa o inversa, en el uso de otros tipos de software o protocolos que permitan mayores límites en la interoperabilidad del hardware, a través de la aplicación de bibliotecas para el procesamiento de imágenes y el uso de diferentes sensores para la detección de objetos.

## CAPITULO IV

### REFERENCIAS

#### 4.1 Referencias Bibliográficas

- [1] A. Valera, A. Soriano, and M. Vallés, “Plataformas de Bajo Coste para la Realización de Trabajos Prácticos de Mecatrónica y Robótica,” *RIAI - Rev. Iberoam. Autom. e Inform. Ind.*, vol. 11, no. 4, pp. 363–376, 2014.
- [2] P. González-Nalda, I. Calvo, I. Etxeberria-Agiriano, E. Zulueta, and J. M. López-Guede, “Hacia un framework basado en ROS para la implementación de Sistemas Ciberfísicos,” *XXXVI Jornadas de Automática*, no. September, pp. 1050–1057, 2015.
- [3] M. V. García, “Desarrollo de Comunicación OPC-UA en Sistemas CPPS de Bajo Costo,” *J. Appl. Microbiol.*, vol. 119, no. 3, p. 859–867, 2015.
- [4] M. M. Cárdenas, P. P. Barrios, K. M. G. Moreno, J. F. S. Arismendy, and M. C. O. Ávila, “Diseño y Construcción del Prototipo de un Brazo Robótico con Tres Grados de Libertad, como Objeto de Estudio,” *Ingeniare*, vol. 0, no. 18, pp. 87–94, 2016.
- [5] K. J. Saumeth C., F. Pinilla T., A. Fernández A., D. J. Muñoz A., and L. A. Cruz S., “Sistema Ciber-Físico de una CNC para la producción de circuitos impresos,” *IV Congr. Int. Ing. Mecatrónica y Autom. CIIMA*, no. January, pp. 154–155, 2015.
- [6] J. Gutiérrez Pérez, “Introducción a ROS en Raspberry Pi,” *Universitat Oberta de Catalunya*, 2017.
- [7] M. V. García, E. Irisarri, and F. Pérez, “Integración Vertical en plantas industriales utilizando OPC UA e IEC-61499 (Vertical Integration in factories using OPC-UA and IEC-61499),” *Enfoque UTE*, vol. 1, no. 1, pp. 287–299, 2017.
- [8] M. V. García, E. Irisarri, F. Pérez, E. Estévez, and M. Marcos, “Arquitectura de Automatización basada en Sistemas Ciberfísicos para la Fabricación Flexible en la Industria de Petróleo y Gas,” *RIAI - Rev. Iberoam. Autom. e Inform. Ind.*, vol. 15, no. 2, pp. 156–166, 2018.

# ANEXOS

Artículo Aceptado para la publicación en la revista Advances in Intelligent Systems and Computing Series Ed.: Kacprzyk, Janusz. Originally published with the title: Advances in Intelligent and Soft Computing ISSN: 2194-5357

Glosario de términos

Carta de Aceptación del Artículo

## Glosario de Términos

**API:** La interfaz de programación de aplicaciones (application programming interface), es un conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos).

**CPS:** Los sistemas ciberfísicos (cyber-physical system) son todos aquellos dispositivos que integran capacidades de computación, almacenamiento y comunicación con el fin de controlar e interactuar con un proceso físico.

**EtherCAT:** Ethernet para el Control de Tecnología (Ethernet for Control of Automation Technology) de Automatización es un protocolo informático de código abierto y alto rendimiento creado y desarrollado por Beckhoff Automation. Pretende utilizar protocolos de Ethernet en un entorno industrial.

**Ethernet:** Es un estándar de redes de área local, define las características de cableado y señalización; de nivel físico y los formatos de tramas de datos del nivel de enlace de datos del modelo OSI.

**PC-ITX:** es un formato de placa base con formato de origen propietario, sus especificaciones son abiertas. De hecho, otros fabricantes tienen productos en este formato.

**ROS:** El Sistema Operativo Robótico (Robot Operating System) es un framework para el desarrollo de software para robots que provee la funcionalidad de un sistema operativo en un clúster heterogéneo.

**SBC:** Una placa computadora u ordenador de placa reducida (Single Board Computer) es una computadora completa en un solo circuito.

**SOEM:** SOEM es una biblioteca maestra EtherCAT escrita en c. Todos los usuarios están invitados a estudiar la fuente para comprender cómo funciona un maestro EtherCAT y cómo interactúa con los esclavos EtherCAT.

21/1/2020

[WorldCist'20] Your submission has been accepted!

**Fecha:** 19-12-25 [23:44:30 CET]  
**De:** WorldCist'20 <worldcist@gmail.com>  
**Para:** mgarcia294@ehu.eus  
**Asunto:** [WorldCist'20] Your submission has been accepted!

Dear Marcelo V. Garcia,

On behalf of the WorldCist'20 - 8th World Conference on Information Systems and Technologies, I am pleased to inform you that your submission #311, titled "Design of a gravity compensation for robot control based on low-cost automation" has been accepted like a Full Paper.

We have included the reviewers' comments at the end of this message.

Please consider reviewers' comments and the rules of edition strictly (<https://www.springer.com/us/authors-editors/conference-proceedings/conference-proceedings-guidelines>) to prepare the camera-ready version of the paper, saved in Word or Latex format and, also, in PDF format. These files must be accompanied by the Consent to Publish form (<http://www.worldcist.org/copyright.pdf>) filled out, in a ZIP file, and uploaded until January 8 at <http://www.aistic.org/wcist2020/oc20/openconf.php> .

Additionally, you also need to make your conference registration until January 8, in order your article can be published and presented. The payment of registrations from countries outside the European Union must be done by debit or credit card through our PayPal system.

Congratulations,

Chairs, WorldCist'20  
<http://www.worldcist.org/>  
worldcist@gmail.com

\*\*\*\*\*

# Design of a gravity compensation for robot control based on low-cost automation

1

**Abstract.** In recent years, robotics applications based on open source have been developed, where the precision in movements is considered as the main goal when following a trajectory, however, due to different circumstances, small errors can interfere with the individual control of the manipulator joints, to improve the accuracy of the end-effector in cartesian space there are different kind of methods as the gravity compensation which is included in the dynamic model. In addition to the development of a system that integrates concepts of Industry 4.0, a low-cost controller is used for this system, where high communication and processing capabilities allow rapid exchange of information between components. In this article is developed an application over Raspberry Pi 3B for control the arm of KUKA youBot through a graphical interface where positions can be saved and reproduces.

**Keywords:** Gravity compensation · arm manipulators · PD control · position control · low-cost automation.

## 1 Introduction

Process control through embedded systems is going through a stage of growth thanks to the opening of industries to systems whose monitoring and control capabilities have improved their mode of operation despite their low cost, the progress of applications based on internet allows distributed production environments through the interaction between integrated computer systems and physical environments, giving rise to automatic physical processes and control of the Cyber Physical Systems (CPS), demonstrating the high potential it has in the productive sector, in addition It should be taken into account that a complete system must have several requirements, which have different applications with their own barriers [9].

The use of Robot Operating System (ROS) as an implementable framework within the Cyber-physical Systems, as it simplifies the communication and

monitoring of the robot control status, this so-called software is installed on GNU/Linux allowing several devices to be incorporated into a distributed network, where a master node allows the rest of the nodes to communicate through TCP/IP, in addition these elements that are part of the CPS architecture can be secured through the implementation of a VPN server, where several advantages of this network of ROS nodes are obtained, such as modularity of the architecture allowing adaptation to other systems, redundancy is possible due to the ROS publication/subscriber scheme, in addition to allowing cross-platform access and remote control through another element connected to the same network [4].

For the control of the joints of the robotic arm two methods have been defined, a change from the current position to the next or a variation in the speed of a joint until reaching the desired position. For both, extra calibration is required considering gravitational compensation, which will allow the monitoring of the final effector trajectories in an appropriate manner reducing the error in position and speed [10].

The aim of this research is to present the design of an advanced control system through gravitational compensation using a low-cost hardware platform of the SBC type that allows programming under the Application Programming Interface (API) of the Kuka youBot robotic arm, which includes methods and classes for the development of programs, in addition with using of ROS, which enable control over open source drivers, leading to interoperability between independent systems.

The content of the article is structured as follows: Section 2 provides brief concepts which will allow a better understanding of the following sections. Section 3 presents the case study used for the research method. While in Section 4, the design and implementation of a gravity compensation for robot control based on low-cost automation are showed. The discussion of results are presented in Section 5. Finally, the conclusions and future work are presented in Section 6.

## 2 State of Technology

### 2.1 KUKA youBot™

It is defined as a mobile manipulator developed primarily for education and research, consisting of two parts that can be used independently, a five degree freedom robotic arm with two fingers as a clamp and an omnidirectional base, the arm's API is compatible with ROS and can be communicated through Ethernet and EtherCAT, and at its base it has a mini ITX PC integrated with its CPU [11].

As shown in Fig. 1, the robotic arm consists of five rotary joints and a two-finger clamp in parallel as a final tool, it reaches a height of 655mm, was designed to carry a load of up to 0.5kg and its rotary joints can be accessed through EtherCAT, joints two, three and four have their rotational movement parallel to one axis, while joints one and five rotate in parallel to another whenever the arm points up, the maximum rotation speed of each joint is 90 deg/s while the final effector can be opened 11.5mm for each finger. The communication protocol



**Fig. 1.** Kuka youBot Arm

defined for the Kuka YouBot uses the standard Ethernet network of the Simple Open Source EtherCAT Master (SOEM) type, which allows the implementation of an open source controller for the master [3].

The youBot driver has a library with classes defined in C++ that allow the control and monitoring of actuators and sensors, it provides the functionalities to establish cartesian position and velocity values, the API provided by the driver contains subsystems for handling the joints where there are classes for the arm and base [2].

## 2.2 Gravity Compensation

The field of stability control of manipulative robots has become more important since its proposal in 1981, as this only applied to robotic arms that only have rotating joints, for mathematical modeling equations are used that are nonlinear, however, Lyapunov's theory is often used as an auxiliary means [8].

The development of the PD control with gravity compensation for manipulators is based on the assumption that the dynamics of the actuators do not exist, so it should be considered that the control signal is not the torque but the voltage that is applied to the motors, where the torque comes as the result of the electrical interaction of the actuators [1].

Without considering external disturbances, the mathematical model of a rigid robot that has a number  $n$  of links is described (See Eq. 1):

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + F(\dot{q}) = \tau \quad (1)$$

Where  $M(q)\ddot{q}$  represents matrix of symmetric manipulator inertia,  $C(q, \dot{q})\dot{q}$  is matrix of centripetal and coriolis forces,  $G(q)$  is gravitational pairs vector,  $F(\dot{q})$  is friction pairs and  $\tau$  represents input torque vector.

The derivative proportional control algorithm with gravity compensation is defined by the correction of the joint positioning error (See Eq. 2):

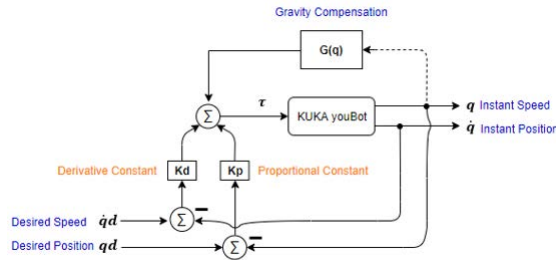
$$\tau = k_p \tilde{q} - k_d(\dot{\tilde{q}}) + G(q) \quad (2)$$



Where  $k_p \tilde{q}$  is the Proportional gain multiplied by the position error,  $k_d(\dot{\tilde{q}})$  is the Derivative gain multiplied by the speed of movement and  $G(q)$  represents the Gravitational torque compensation.

Gravitational compensation is defined as a term proportional to the position error, to which a damping consisting of the speed of movement is added, this part of the control consists of a mechanical retention that absorbs the impulses resulting from the response, suppressing peaks and oscillations, for the stationary stage it is estimated that the position of the effector or terminal tool is constant, assuming a value of zero at the speed of movement [5].

According to what is expressed in [6], the control loop with gravity compensation for robots is established as an equation that needs a constant measurement of position and speed, Eq. 2, determines the dynamic model the dynamic model for PD control with gravitational compensation for manipulative robots.



**Fig. 2.** Dynamic Model for PD control with gravitational compensation

Additionally, Fig. 2 shows the block diagram equivalent to the PD control law with gravity compensation, where  $K_p$  and  $K_d$  are defined as positive symmetric matrices, however this type of optimization requires a partial knowledge of the manipulator, considering values of length, mass and rotational parameters of the components of the robotic arm.

### 2.3 ROS

It is an open source workspace that facilitates and speeds up the construction of robotic applications such as message delivery, package management and hardware device drivers. Originally designed in order to simplify distributed systems, as seen in Fig. 3, the building blocks of a ROS application are defined as nodes, where each is an entity that can communicate with other nodes that are part of an operating system process, the nodes supported by ROS are mainly written in C++ and Python. Nodes in a ROS environment can communicate in three ways such as: Posting messages to a topic, listening to messages posted on a topic and calling a service provided by another node [7].

Initially for the management of the communication system, a master ROS must be initiated, which is responsible for allowing the nodes to meet and ex-

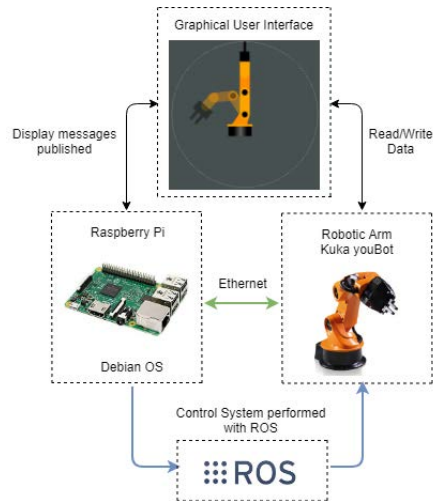


**Fig. 3.** Component system of a ROS model

change messages, the use of topics or topics is defined as logical connection paths, where the nodes can publish topics or subscribe to them.

### 3 Study Case

The positioning of objects is one of the most common tasks performed by robotic arms, this type of procedure requires a control that allows the rotary joints to be manipulated according to the need of the process.



**Fig. 4.** Study Case Architecture

The present work shows the design of a control system for the robotic arm of the Kuka youBot manipulator developed on the Raspberry Pi 3B embedded card, this for the realization of a "pick and place" positioning task and the monitoring through the interface chart which has the option to save and reproduce positions, as seen in Fig. 4, the architecture is generated through the integration and cross-platform work of ROS and the youBot driver, allowing the development of libraries, topics and nodes, on which the communication data and messages will be sent and received through the Ethernet protocol.

The graphical interface allows control by sending position and velocity values through the robotic arm driver, for which classes and methods that allow data to be sent to the rotary joints are developed in C++, in addition the manipulation through gravitational compensation control, whereby the controlled delivery of current to the joint motors ensures that there is no damage to the manipulator joints.

In the present investigation it is used low-cost SBC type embedded cards because are considered as controllers or potential solutions to data processing and monitoring problems, being compatible with open source, the control is designed with gravitational compensation, which consists of the interpolation of the sent positions, so that the current values sent to the motors are regulated proportionally and derivatively, in order to make the teaching of positions more versatile this control method is included, this allows an operator handle the robotic arm according to your need manually.

## 4 Proposed Implementation

### 4.1 Gravitational Compensation

A new ROS node is established that will be mainly responsible for the calibration and control of the robotic arm, for this the physical parameters required by the gravitational compensation module such as friction, mass, gravity, etc. have been previously configured.

When the gravitational compensation module is started, the above mentioned parameters are loaded, simultaneously and automatically the initialized robotic arm calibration is performed, each rotary joint is defined as an EtherCAT slave, and for each of them iterations are performed to find the most exact value in which the torque made by the motors is reduced, this is because the torque changes are proportional to those of current.

Mainly based on the use of the nlopt library, developed for non-linear optimization systems, this allows the evaluation of several algorithms for long calibration routines, allowing several parameters to be tested in a short period of time.

When the module is activated from the GUI, the current values delivered to the joints are the minimum so that it can maintain the links statically, therefore, when applying an external force in order to manipulate the robotic arm, it does not present opposition, for which the operation of the control is verified by gravitational compensation.

### 4.2 Graphical User Interface

The GUI developed for the interactive control of the robotic arm is presented as a ROS node, which exchanges data with the controller node and this in turn is responsible for the control of the hardware device through a third node implemented with the robot in URDF format, the interface is developed under

C++ and consists of discrete buttons for the execution of movements through the sending of positional values or by velocity variation, as well as the activation of the control module for gravitational compensation, it also has the save function and load of states, so that the user can manually perform the positioning tasks raised in the investigation.

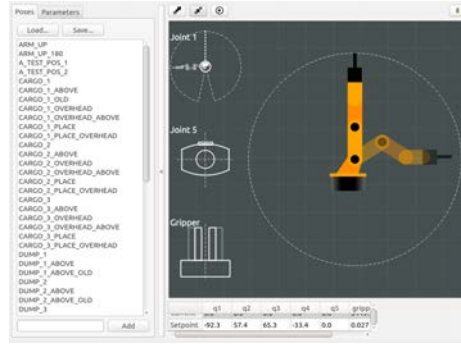


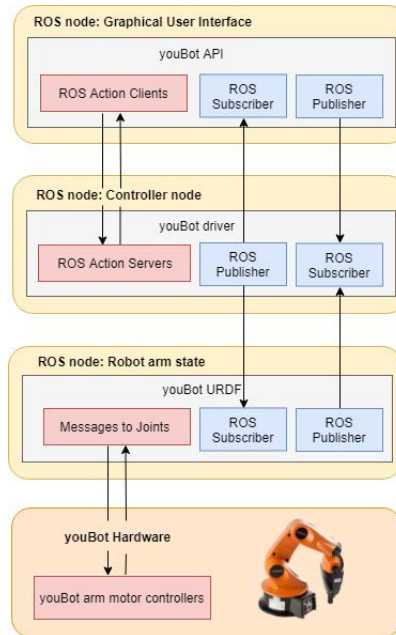
Fig. 5. Graphical User Interface

Within the GUI, in addition to the palette to load and save arm states, as shown in the Fig. 5, a 2D visualization of the robotic arm can be visualized, where two stages of this one are presented, a current one and one that will understand the objective to which the arm is to be moved, for the understanding of the possible movement made in the missing plane, three views are included that correspond to the upper detail of the first joint, the joint number 5 for the rotation of the gripper and a front of the latter to correctly visualize the opening or closing of the same. In the lower part, the current rotational parameters of the 5 joints are observed plus the longitudinal variation of the gripper, in order to maintain the positional values within the allowed ranges of each motor.

### 4.3 ROS System

For the implementation of the initially proposed study case, a ROS system with three nodes is developed as seen in Fig. 6, which will allow control over the robotic arm of the Kuka youBot manipulator through an advanced method based on gravitational compensation, if the three nodes are going to be subscribers and publishers of different topics, this in order that the messages sent from the developed GUI arrive through the controller node to the physical rotary meetings, within the GUI the actions to be executed by each component are detailed through files in C++ of the same, it is described through the youBot API how the message will be published and read by the controller node, which is the core of the complete system, this will initialize the ROS master.

In addition to executing the robotic arm driver with its respective configurations and mainly the gravitational compensation module with its certain cali-



**Fig. 6.** ROS node Architecture

bration parameters, the node transmits the messages sent from the GUI through the robot in URDF format, which allows in a third node access to the hardware device through the EtherCAT protocol, this last node describes the geometry of several physical elements, with which the driver ensures that the initialized robot is the correct one for the API used, translating the messages sent to the motors as positional, velocity or torque values in current variations or specific limits.

The ROS Rqt Graph tool is used to get a real-time diagram of the iteration between nodes, topics and messages, where you can observe the flow of information within the developed ROS system, the type of messages used to provide control over rotary joints of the robotic arm allow to control the motors through the youBot driver, in Fig. 7, the graph obtained is presented, which is equivalent to the system proposed as a case study.

The type of messages sent from the controller node to the node of the URDF robot, are `sensor_msgs /JointState`, these are used to transmit values in float64 format within vectors, for this a vector of 5 positions is defined, where each of them will be occupied by each of the rotary joints of the robotic arm, this corresponds to the 5 degrees of freedom and is established by the youBot API, these spaces will be filled when a position, velocity or effort type value is sent to the array, when a position has not been filled, a value of zero is assumed and this does not affect the execution of the order sent to the robotic arm.

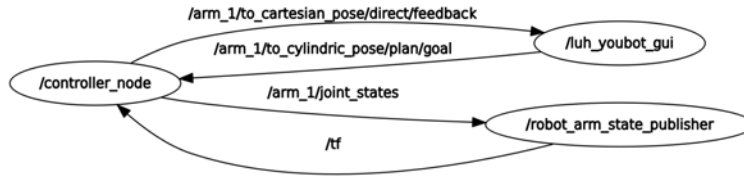


Fig. 7. Rqt Graph

## 5 Discussion of Results

To check the operation of the developed controller node, all measured values greater than 0.01sec are established as a delay, the measurements are defined in the executable file of the mentioned node, where the difference between the established frequency and the operation of the robotic arm is calculated, the delay in the operation of the controller as a function of time is obtained from the inverse value of the result.

The case study presents a positioning task called "pick and place", for which the gravitational compensation module was used for the manual control of the robotic arm towards the desired positions, in addition, the saving and reproduction of positions was used through the GUI To check the operation of the same, finally two positions were obtained for the development of the task.

As can be seen in Fig. 8, which is a graphical representation of the delays obtained as a function of time, the controller does not have large delays in the execution of the gravitational compensation module, since these remain close to the established limit of 0.01sec, except for extreme cases where values suffer relatively high peaks.

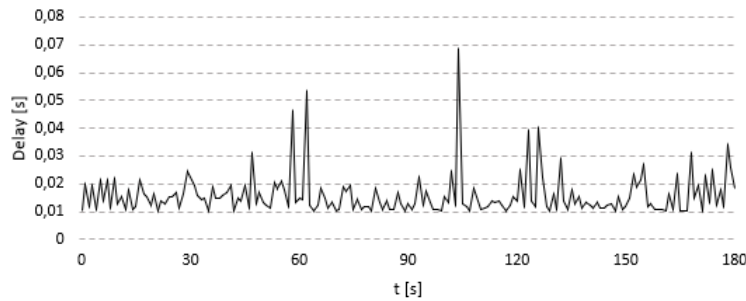


Fig. 8. Delays get in the study case

## 6 Conclusions and Future Work

This work shows the design of a controller node within a system developed through the ROS API and the youBot driver, mounted on a low-cost SBC card, the proposed architecture as a case study raises interoperability between hardware platforms based on open source, the elaborated application is considered as a metapackage which contains the packages and files necessary for the control of the robotic arm of the Kuka youBot manipulator through EtherCAT. The calibration and control tasks of the rotary joints of the robotic arm for control through the gravitational compensation module, are performed in the background and automatically once the controller node has been initialized, when this process ends, it can be executed the GUI developed, which allows a more intuitive control for the user, since it allows to observe the current position and the objective to which you want to move, in addition to making possible the selection of the type of control, either by position, velocity or by the gravitational compensation module.

The controller works constantly since the ROS master and the entire system was executed, for a correct operation a value of 200Hz must be established for the frequency of the robotic arm, which varies for each joint thanks to the actual operating parameters, this creates an existing delay in the control loop, which is shown in the terminal and allows observing as a function of time how much the tracking of the developed pick and place process is delayed, finally obtaining low values that vary according to the accuracy in the positions that must be reached by the robotic arm.

Future lines of research should focus on the development of different control methods based on the prediction of trajectories through direct or reverse kinematics, on the use of other types of software or protocols that allow greater limits on the interoperability of the hardware, through the application of libraries for image processing and the use of different sensors for object detection.

## References

1. Binazadeh, T., Yousefi, M.: Designing a cascade-control structure using fractional-order controllers: Time-delay fractional-order proportional-derivative controller and fractional-order sliding-mode controller. *Journal of Engineering Mechanics* **143**(7), 04017037 (2017). [https://doi.org/10.1061/\(ASCE\)EM.1943-7889.0001234](https://doi.org/10.1061/(ASCE)EM.1943-7889.0001234)
2. Di Napoli, G., Filippeschi, A., Tanzini, M., Avizzano, C.A.: A novel control strategy for youbot arm. In: *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*. pp. 482–487 (Oct 2016). <https://doi.org/10.1109/IECON.2016.7793658>
3. Garcia, C.A., Franklin, S.L., Mariño, C., Villalba, W.R., Garcia, M.V.: Design of flexible cyber-physical production systems architecture for industrial robot control. In: *2018 IEEE Third Ecuador Technical Chapters Meeting (ETCM)*. pp. 1–6 (Oct 2018). <https://doi.org/10.1109/ETCM.2018.8580338>
4. Garcia, C.A., Lanás, D., Edison, A.M., Altamirano, S., Garcia, M.V.: An approach of cyber-physical production systems architecture for robot control. In: *IECON*

- 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society. pp. 2847–2852 (Oct 2018). <https://doi.org/10.1109/IECON.2018.8591286>
5. de Gea Fernández, J., Mronga, D., Günther, M., Knobloch, T., Wirkus, M., Schröer, M., Trampler, M., Stiene, S., Kirchner, E., Bargsten, V., Bänziger, T., Teiwes, J., Krüger, T., Kirchner, F.: Multi-modal sensor-based whole-body control for human–robot collaboration in industrial settings. *Robotics and Autonomous Systems* **94**, 102 – 119 (2017). <https://doi.org/https://doi.org/10.1016/j.robot.2017.04.007>, <http://www.sciencedirect.com/science/article/pii/S0921889016305127>
  6. Hernández, V., Santibáñez, V., Carrillo, R., Molina, J., López, J.: Control pd de robots: Dinámica de actuadores y nueva sintonía. *Revista Iberoamericana de Automática e Informática Industrial RIAI* **5**(4), 62 – 68 (2008). [https://doi.org/https://doi.org/10.1016/S1697-7912\(08\)70178-X](https://doi.org/https://doi.org/10.1016/S1697-7912(08)70178-X), <http://www.sciencedirect.com/science/article/pii/S169779120870178X>
  7. Koubâa, A.: *Robot Operating System (ROS)*. Springer (2017). <https://doi.org/https://doi.org/10.1007/978-3-319-26054-9>
  8. Peidró, A., Reinoso, Ó., Gil, A., Marín, J.M., Payá, L.: Análisis de estabilidad de singularidades aisladas en robots paralelos mediante desarrollos de taylor de segundo orden (2017)
  9. Wang, L., Törngren, M., Onori, M.: Current status and advancement of cyber-physical systems in manufacturing. *Journal of Manufacturing Systems* **37**, 517 – 527 (2015). <https://doi.org/https://doi.org/10.1016/j.jmsy.2015.04.008>, <http://www.sciencedirect.com/science/article/pii/S0278612515000400>
  10. Yu, C., Li, Z., Liu, H.: Research on gravity compensation of robot arm based on model learning\*. In: 2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM). pp. 635–641 (July 2019). <https://doi.org/10.1109/AIM.2019.8868673>
  11. Zhang, B., Gao, S.: Kuka youbot arm path planning based on gravity. In: 2018 International Conference on Mechanical, Electrical, Electronic Engineering Science (MEEES 2018). Atlantis Press (2018/05). <https://doi.org/https://doi.org/10.2991/meees-18.2018.75>, <https://doi.org/10.2991/meees-18.2018.75>