UNIVERSIDAD NACIONAL DEL CALLAO

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

TESIS

"MODELAMIENTO Y ANÁLISIS DE UN ROBOT DE 3 G.D.L PARA LA COMPRENSIÓN DE ALGORITMOS DE CONTROL, EN ESTUDIANTES DE INGENIERÍA ELECTRÓNICA"

PARA OBTENER EL TITULO PROFESIONAL DE INGENIERO ELECTRÓNICO

PRESENTADO POR LOS BACHILLERES:

- ✓ TALAVERA BENANCIO, CÉSAR CHRISTIAN
- ✓ LARA FIGUEROA, RONALD MICHAEL
- ✓ QUIROZ ARANIBAR, CARLOS ALBERTO

Callao, 2017

PERU

UNIVERSIDAD NACIONAL DEL CALLAO

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

TESIS

"MODELAMIENTO Y ANÁLISIS DE UN ROBOT DE 3 G.D.L PARA LA COMPRENSIÓN DE ALGORITMO DE CONTROL, EN ESTUDIANTES DE INGENIERIA ELECTRÓNICA"

PARA OBTENER EL TITULO PROFESIONAL DE INGENIERO ELECTRÓNICO

PRESENTADO POR LOS BACHILLERES:

- **TALAVERA BENANCIO CESAR CHRISTIAN.**
- **LARA FIGUEROA RONALD MICHAEL.**
- **QUIROZ ARANIBAR CARLOS ALBERTO.**

ASESOR: Mg. Ing. MOSCOSO SÁNCHEZ JORGE ELÍAS

CALIFICACION: 14 (CATORCE)

Ing. LUIS E. CRUZADO MONTAÑEZ	**Ing. CARLOS A. MORENO PAREDES**
PRESIDENTE DE JURADO	**SECRETARIO**

Ing. ABILIO B. CUZCANO RIVAS

VOCAL

Callao, PERU 2017

AGRADECIMIENTOS

Este Trabajo de Tesis es el fruto de todas las enseñanzas impartidas por todos nuestros profesores a lo largo de la carrera de Ingeniería.

Cesar Talavera:

Gracias a mi padre y a mi madre por su apoyo incondicional a lo largo de todos estos años de mi carrera, a mis hermanos por siempre haber sido una fuente de apoyo en momentos de los altibajos de la vida. A mis compañeros Ronald y Carlos por ser los más importantes en la finalización de este trabajo.

Ronald Lara:

A mi madre por todo el apoyo recibido .Su esfuerzo, paciencia, dedicación y sacrificio hicieron de este sueño, un logro. A mi esposa, que me acompaño en momentos buenos y malos, a mi hijo por ser la fortaleza y alegría que necesita para salir adelante.

Carlos Quiroz:

Quiero agradecer primero a Dios que nos iluminó y dio fortaleza en los momentos de dificultad. A mis compañeros de tesis que tuvieron siempre la paciencia suficiente y la entrega necesaria para culminar con este trabajo. Agradezco a mi padre y a mi madre que siempre me brindaron su comprensión y apoyo.

INDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente trabajo que describe Modelamiento y análisis de un robot de 3 G.D.L para la comprensión de algoritmos de control, en estudiantes de ingeniería electrónica.

En primer lugar, en el Marco Teórico, se realiza una muy breve introducción a los sistemas robóticos. Seguidamente, se exponen los fundamentos matemáticos con los que se obtiene el modelo cinemático, así como distintos tipos de trayectoria. Para comprender mejor las condiciones reales del control cinemático, se explica el desarrollo del modelo Dinámico, según las dimensiones de nuestro prototipo a modelar.

En segundo lugar se presenta el diseño mecánico de un robot para fines educativos y las adaptaciones electrónicas que se hicieron para establecer una comunicación con la pc y poder enviar órdenes al robot y observar de esa forma el seguimiento de trayectoria en un prototipo real.

Como parte final se somete a prueba el controlador PID creado en el entorno de Simulink & Matlab con el robot real, observándose ligeros picos de distorsión a la hora de comparar el movimiento de sus articulaciones y el seguimiento de las trayectorias ordenadas a seguir. para corregir ello se utiliza el método de sintonización de **Ziegler – Nichols"** para sintonizar las variables del controlador PID.

INTRODUCCION

Nuestra sociedad se dota cada día de un entramado tecnológico mayor. Los sistemas automáticos rompieron las barreras que los limitaban a la gran producción fabril, las telecomunicaciones hace mucho que dejaron de limitarse a los proyectos militares y aeroespaciales, para formar una parte de importancia creciente en nuestras vidas.

Los sistemas robóticos no podían quedarse atrás; año tras año los robots van ocupando más parcelas de nuestra actividad cotidiana en todos los ámbitos y van superando las rigideces de construcción y programación, así como el alto coste.

Sin duda alguna todos los avances que ha tenido la tecnología en estos años no habrían podido darse sin el mejoramiento constante de los algoritmos de control que permitieron la creación de controladores capaces de automatizar dispositivos y sistemas industriales, por ello el desarrollo de esta Tesis se enfocara en brindar un modelo que describa el comportamiento de un de un robot de 3 grados de libertad y de cómo el algoritmo de control PID permite mediante la sintonización de sus variables, un mejor seguimiento a las trayectorias ordenadas mediante el uso de trayectorias creados en el entorno MATLAB & SIMULINK.

I. PLANTEAMIENTO DEL PROBLEMA

1.1. DETERMINACIÓN DEL PROBLEMA

Para el desarrollo de esta Tesis escogeremos un prototipo de robot de 3 grados de Libertad, cuyo diseño ya es conocido y que usaremos para plantear nuestro Modelamiento y análisis. Basándonos en lo dicho el principal problema que surge aquí es lograr que el efector final llegue a su posición deseada.

1.2. FORMULACIÓN DEL PROBLEMA

Específicamente se pretende responder a la pregunta: **¿Qué se investigará?,** a través del Modelamiento del robot, se debe determinar su comportamiento mediante las cinemáticas y dinámicas que actúan en cada una de las articulaciones al fin de lograr un mejor control de trayectoria con la aplicación de algoritmos de control.

1.3. OBJETIVOS DE LA INVESTIGACIÓN

1.3.1. OBJETIVO GENERAL

El objetivo general de este trabajo es brindar un modelado y análisis de un brazo robótico de 3 grados de libertad para la comprensión de algoritmos de control en estudiantes de Ingeniería Electrónica, primero solo usando el entorno de SIMULINK aplicando cinemática directa e inversa y su respectivo modelo Dinámico del modelo, para luego explicar la sintonización PID sobre un modelo de robot real ya realizado [autor] y cuyo diseño a nivel de hardware y software será adjuntado en el desarrollo de esta Tesis, De esa manera se espera que los alumnos Noten mediante la adquisición de datos en el entorno SIMULINK de MATLAB, el comportamiento de las articulaciones de un robot real y como esta mejora cuando sintonizamos el controlador PID.

1.3.2. OBJETIVOS ESPECÍFICOS

- La implementación de la cinemática directa y cinemática inversa mediante el uso de interfaces en MATLAB.
- Realizar un controlador PID para el seguimiento de trayectorias.
- Utilizar un método de sintonización para el controlador PID que regulara los movimientos de las articulaciones en el robot didáctico.

1.4. JUSTIFICACIÓN DE LA INVESTIGACIÓN

El estudio se viabiliza y justifica por sí solo en la medida que este trabajo permita que el estudiante comprenda la viabilidad de la sintonización de los algoritmos de control (PID) con un ejemplo práctico y de aplicación directamente ligada a la carrera como es la rama de Robótica para ello primero el alumno tiene que comprender el análisis de la cinemática directa e inversa y el desarrollo de trayectorias del robot mediante su modelado dinámico usando para ello un entorno de simulación en MATLAB & SIMULINK, poniendo luego a prueba la sintonización PID en un robot real con fines didácticos.

II. MARCO TEORICO

2.1. ANTECEDENTES DEL ESTUDIO

A medida que hay avances tecnológicos, la investigación científica requiere de equipo capaz de responder a las necesidades y exigencias de los investigadores, ingenieros y estudiantes que día con día trabajan en ciencia y tecnología aplicada. Para el campo de la Robótica principalmente se requieren estudiantes para el manejo de hardware y software especializado en la adquisición de datos que les permita fortalecer conocimientos del área de robots industriales.

Es de suma importancia que los algoritmos desarrollados en laboratorios de investigación sean probados en un equipo que permita validar teorías, por lo que se busca tener plataformas de fácil implementación. A su vez, por motivos prácticos, es conveniente el uso de equipo existente y su modernización mediante software y hardware actualizado.

2.1.1. ¿Qué es un robot?:

"Un robot es una máquina o ingenio programable, capaz de manipular objetos y realizar operaciones antes reservadas solo a personas" (DRAE, 2014).

2.1.2 Tipos de robots:

Los robots pueden ser clasificados según su ámbito de aplicación en:

-Robots industriales: Según la norma ISO 8373 (2012), un robot industrial se define como aquel robot controlado automáticamente, reprogramable, con múltiples aplicaciones, manipulador, programable en 3 o más ejes, que puede ser o bien fijado en un sitio o móvil para su utilización en aplicaciones de automatización industrial.

-Robots de servicio: Según la misma norma, un robot de servicio es aquel que realiza tareas útiles para los humanos o equipamiento, excluyendo las aplicaciones de automatización industrial.

Otra posible clasificación está relacionada con la movilidad del robot:

-Robots manipuladores: Aquellos que tienen fijo uno de sus extremos. Son, por ejemplo, los brazos robot, y se corresponden con la gran mayoría

de robots industriales; aunque también hay robots de servicio manipuladores.

-Robots móviles: Son robots que pueden desplazarse por sí mismos. Generalmente se utilizan en aplicaciones tanto industriales como de servicio relacionado con el transporte o la reproducción de formas de actuar humanas.

2.1.3 Robot Modelado en esta Tesis.

En el presente trabajo se ha modelado un robot de tamaño mediano de 3 grados de libertad donde todas sus articulaciones son del tipo rotacional, según las dimensiones del robot real que usaremos al final para establecer el control de trayectoria vía MATLAB & SIMULINK. De forma que el ámbito de aplicación para este sería la docencia por lo que podría encuadrarse como un robot de servicio-manipulador.

2.2. Análisis Cinemático

Para analizar el comportamiento de un robot, el primer paso es la obtención de los modelos cinemáticos, el cual, proporciona información sobre el movimiento del robot considerando Únicamente sus características geométricas e ignorando las fuerzas y pares que lo generan, es conveniente para ello conocer los sistemas de referencia de Rotación, Traslación y Transformación Homogénea.

2.2.1 sistema de referencia: rotación, traslación y transformación homogénea:

Para resolver el problema cinemático, a cada elemento le corresponderá un sistema de referencia.

Para comprender mejor las herramientas que se utilizarán a continuación, conviene primero detenerse para comprender la modelización matemática de la rotación y la traslación, para lo cual se van a combinar ejes de referencia fijos con ejes de referencia móviles, y se estudiará su relación.

- Rotación:

Se procede a estudiar las matrices de rotación partiendo de un caso concreto: Sean dos sistemas de referencia: uno fijo (OXYZ) y uno móvil (OUVW), que son inicialmente coincidentes.

En un instante cualquiera, el sistema móvil rota en el sentido contrario a las agujas del reloj alrededor del eje U, de modo que el eje V y el eje W giran un ángulo α respecto a su posición inicial, esto es, respecto al eje Y y al eje Z, respectivamente, como se muestra en la figura 2.1:

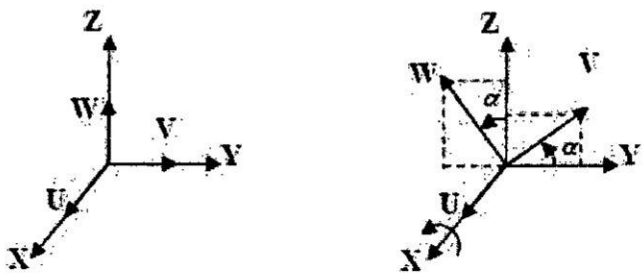


Figura 2.1 rotación de un sistema respecto de otro

En el instante posterior a la rotación del sistema móvil, los vectores unitarios de los ejes de ambos sistemas serían:

$$\vec{\imath}_x = (1,0,0) \quad \vec{\jmath}_y = (0,1,0) \quad \vec{k}_z = (0,0,1)$$

$$\vec{\imath}_u = (1,0,0) \quad \vec{\jmath}_v = (0,\cos\alpha,\sin\alpha) \quad \vec{k}_w = (0,-\sin\alpha,\cos\alpha)$$

Mediante una matriz que recoja en cada elemento la proyección de un eje del sistema de referencia fijo sobre un eje del sistema de referencia móvil (productos escalares de dichos vectores), es posible representar la rotación sufrida por el sistema móvil respecto del fijo. Por este motivo esta matriz recibe el nombre de matriz de Rotación (R):

$$R = \begin{pmatrix} \vec{\imath}_x.\vec{\imath}_u & \vec{\imath}_x.\vec{\jmath}_v & \vec{\imath}_x.\vec{k}_w \\ \vec{\jmath}_y.\vec{\imath}_u & \vec{\jmath}_y.\vec{\jmath}_v & \vec{\jmath}_y.\vec{k}_w \\ \vec{k}_z.\vec{\imath}_u & \vec{k}_z.\vec{\jmath}_v & \vec{k}_z.\vec{k}_w \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix}$$

Esto es extrapolable también a un giro alrededor de **V** y de **W**. Generalizando más, estas Matrices de Rotación tienen unas propiedades comunes:

·Cada columna de la matriz se corresponde con un vector unitario de un eje del sistema móvil puesto en función de los vectores unitarios de los ejes del sistema fijo ($\vec{i}_u$, primera columna, $\vec{j}_v$, segunda columna, $\vec{k}_w$, tercera columna).

·Cada fila de la matriz se corresponde con un vector unitario de un eje del sistema fijo puesto en función de los vectores unitarios de los ejes del sistema móvil ($\vec{i}_x$, primera fila, $\vec{j}_y$, segunda fila, $\vec{k}_z$, tercera fila).

·Los vectores fila y columna son unitarios.

·El determinante de la Matriz de Rotación es 1.

·Los productos escalares entre una fila con otra fila, o entre una columna con otra columna, son nulos, debido a que representan vectores ortogonales unos con otros.

. $R^{-1} = R^T$

. Se pueden multiplicar matrices de rotación para representar una secuencia de rotación finita. La multiplicación es no conmutativa:

-Pre multiplica si gira con respecto a OXYZ. ($R = R_{base\ xyz} \cdot R$)

-Post multiplica si gira con respecto a OUVW. ($R = R \cdot R_{base\ uvw}$)

-Traslación:

La traslación de un punto P, ubicado en un sistema coordenado mediante un vector $\vec{v}$ es:

$$\vec{P}_{xyz} = \vec{v} + \vec{P}_{uvw}$$

-Transformación homogénea:

Para representar en una matriz tanto rotación como traslación, se introduce la Matriz de Transformación Homogénea T, que representa la orientación y posición del sistema OUVW rotado y trasladado con respecto al sistema OXYZ.

$$T = \begin{bmatrix} R_{3x3} & p_{3x1} \\ f_{1x3} & \omega_{1x1} \end{bmatrix} = \begin{bmatrix} rotacion & posición \\ perspectiva & escalado \end{bmatrix}$$

En robótica la transferencia de perspectiva es nula, así que el término correspondiente de la matriz, queda anulado, y ω, que representa el escalado global, será 1, de modo que T queda:

$$T = \begin{bmatrix} R_{3x3} & p_{3x1} \\ 0_{1x3} & 1_{1x1} \end{bmatrix}$$

De forma que se puede representar la posición de un punto P respecto del sistema fijo OXYZ como el producto de la matriz T por las coordenadas de P respecto del sistema móvil OUVW, tan solo introduciendo una coordenada homogénea, ω, que como ya se ha visto, toma el valor 1.

De este modo:

$$\vec{P}_{XYZ} = T \cdot \vec{P}_{UVW} \; con \; \vec{P}_{xyz} = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} y \; \vec{P}_{UVW} = \begin{pmatrix} u \\ v \\ w \\ 1 \end{pmatrix}$$

Con la misma idea en mente, es posible utilizar la matriz T para conocer la posición y orientación final del punto P, $\vec{P}_{XYZ'}$, tras ser rotado y trasladado Respecto de su posición inicial referida al sistema de coordenadas fijo

OXYZ, $\vec{P}_{xyz}$.

De este modo:

$$\vec{P}_{XYZ'} = T \cdot \vec{P}_{xyz}$$

En este caso el vector p_{3x1} inserto en la matriz T representara el vector de traslación.

2.2.2- Aproximaciones al problema cinemático:

Se distingue entre dos formas de plantear el problema cinemático:

-Aproximación activa: El problema de posición se aborda con desplazamientos de los elementos que componen el robot desde una posición de referencia.

-Aproximación pasiva: El problema de posición se plantea a partir de relaciones entre sistemas de referencia asociados a las barras del robot.

Dado que se opta por esta segunda opción, la aproximación pasiva, se desarrollará en mayor profundidad:

La idea fundamental de este método es que un mismo punto tiene diferentes coordenadas en distintos sistemas de referencia, y que

combinando transformaciones se puede pasar la representación del vector de un sistema a otro adyacente, mediante la pre-multiplicación de matrices de transformación homogénea, en la línea de lo visto con anterioridad.

Se define A_n^{n-1} como la matriz de transformación homogénea que nos permite pasar de la representación de las coordenadas de un punto respecto del sistema de referencia n-1 al sistema de referencia n.

Se cumple que: $A_{n-1}^{n-2} \cdot A_n^{n-1} = A_n^{n-2}$

Esta propiedad puede aprovecharse para concatenar distintos sistemas de referencia, ventaja que, como ya puede intuirse, será muy útil en la aproximación pasiva al problema cinemático.

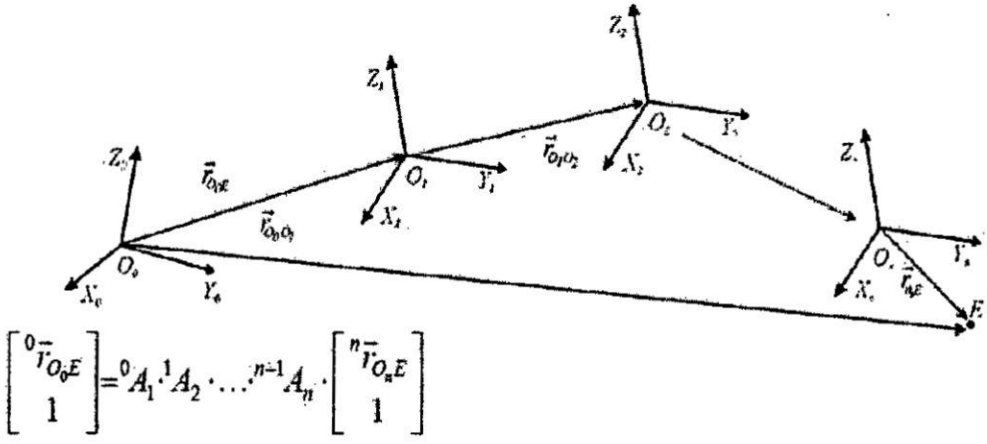


Figura 2.2 Transformación homogénea y sistemas de coordenadas.

2.2.3 CINEMÁTICA DIRECTA

Para determinar las ecuaciones cinemáticas, se utilizará el método matricial propuesto por Denavit y Hartenberg en 1955. El objetivo que tiene la cinemática directa es poder determinar la posición del efector final respecto a un sistema base, con ecuaciones en función únicamente de las variables articulares del robot, mientras que la cinemática inversa es útil para determinar las variables articulares cuando se conoce la posición del efector final respecto a un sistema base.

El establecimiento de los sistemas coordenados se asignan en base a la convención del algoritmo de Denavit-Hartenberg como se observa en la Figura 2.3.

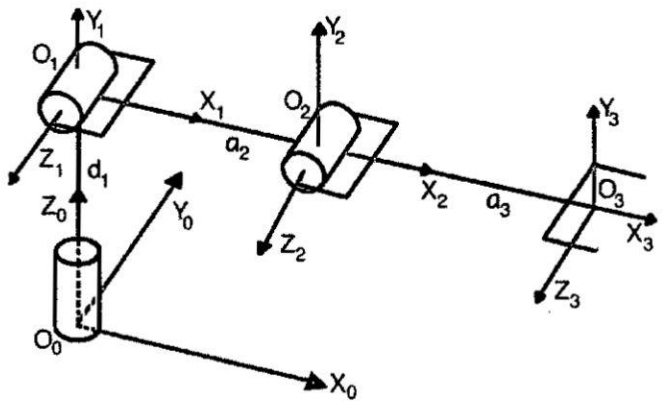


Figura 2.3: sistemas coordenados y ángulos de las articulaciones

Mediante la asignación de los sistemas coordenados, se obtienen los parámetros que van a representar la movilidad de cada articulación del robot. La obtención de los parámetros se observa en la Tabla 1.

Eslabón	a_i	α	d_i	θ_i
1	0	$\pi/2$	d_1	θ_1
2	a_2	0	0	θ_2
3	a_3	0	0	θ_3

Tabla 2.1 parámetros Denavit - Hartenberg del robot

Donde,

- θ_i: Es el ángulo medido del eje X_{i-1}al eje X_i teniendo como eje de giro a Z_{i-1}.
- d_i es la distancia del sistema coordenado $i-ésimo_{-1}$ hasta la intersección de los ejes Z_{i-1} y X_i medida sobre Z_{i-1}.
- a_i es la distancia del sistema coordenado i-ésimo hasta la intersección de Z_{i-1} y x_i medida sobre x_i.

- α_i es el ángulo medido del eje Z_{i-1} al eje Z_i, tomando como eje de giro al eje x_i.

Tomando en cuenta las restricciones de la representación de Denavit-Hartenberg se puede representar cualquier desplazamiento o rotación con las transformaciones homogéneas
A_i de cada eslabón:

$$A_i = \begin{bmatrix} {}^{\circ}R_i & {}^{\circ}O_i \\ 0 & 1 \end{bmatrix} \tag{2.1}$$

Donde, ${}^{\circ}R_i$ es una matriz de rotación, que representa la rotación del sistema coordenado i- ésimo, respecto al sistema base y ${}^{\circ}O_i$ representa la traslación del sistema coordenado i-ésimo, respecto al sistema base. Estas matrices son necesarias para obtener la matriz de inercia del robot.

La matriz de transformación homogénea para cada uno de los sistemas coordenados debe seguir la siguiente ecuación:

$$i-1_{A_i} = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

Durante el desarrollo de este trabajo se obtuvieron las matrices homogéneas del robot, están dadas por:

$$0_{A_1} = \begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 & 0 \\ \sin\theta_1 & 0 & -\cos\theta_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$1_{A_2} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & a_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & a_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$2_{A_3} = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & a_3\cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & a_3\sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

De (2.4) se obtienen las ecuaciones para la Cinemática Directa:

$$X = a_3C\theta_1C\theta_2C\theta_3 - a_3C\theta_1S\theta_2S\theta_3 + a_2C\theta_1C\theta_2 \quad (2.6)$$

$$X = a_3S\theta_1C\theta_2C\theta_3 - a_3S\theta_1S\theta_2S\theta_3 + a_2S\theta_1C\theta_2 \quad (2.7)$$

$$Z = a_3S\theta_2C\theta_3 + a_3C\theta_2S\theta_3 + d_1 + a_2S\theta_2 \quad (2.8)$$

Donde, x, y, z son la posición del efector final sobre el sistema coordenado base (X_0,Y_0,Z_0).

Para poder hacer el análisis del robot se obtuvo el par máximo de cada uno de los motores que actúan en las articulaciones, esto depende del tamaño de cada eslabón, los cuales son los que se tomarán en cuenta para el diseño del robot, por lo que d_1: 23 [cm], a_2: 20 [cm] y a_3: 13 [cm].

2.2.4 Cinemática Inversa.

El objetivo de la cinemática inversa es determinar las variables articulares en función únicamente de la posición del efector final. En general, es más complicado que la cinemática directa, ya que no siempre puede obtenerse una solución analítica cerrada a este problema. Sin embargo, se puede obtener una solución para el problema cinemático inverso mediante un enfoque geométrico.

Para obtener las variables articulares es conveniente hacer proyecciones sobre planos para comprender su comportamiento. En este sentido, la solución para θ_1 resulta sencilla de apreciar, así que la primer variable articular que se obtendrá es θ_1. Para esto basta con hacer una proyección de x, y, z, (Posición del efector final) sobre el plano X_0 y Y_0, nótese que no hay relación del efector en Z_0 con θ_1.

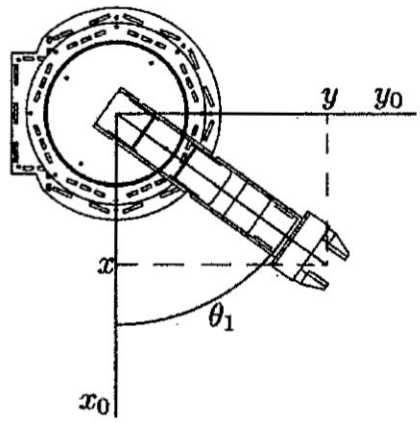


Figura 2.4: Vista del plano X_0 y Y_0, para la obtención de θ_1

De la Figura 2.4 se obtiene usando los comandos que usamos en MATLAB:

$$\theta_1 = \text{atan2}(y, x) \qquad (2.9)$$

Para poder determinar θ_2 y θ_3 se hace una proyección sobre el plano $Z_1;Y_1$, tal que sea más evidente la trayectoria que pueden describir los eslabones del manipulador. Es importante mencionar que se obtendrá primero θ_2, ya que θ_3 está en función de θ_2.

"atan2 es un comando de MATLAB que expresa arcotangente en el plano cartesiano"

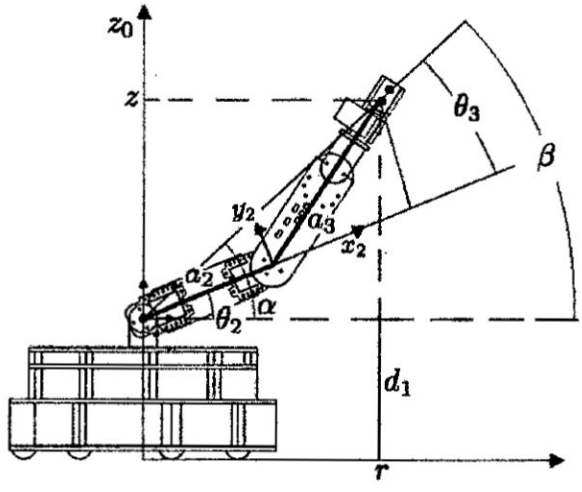


Figura 2.5: Proyección sobre el plano Z_0 y Y_0, para mostrar la obtención de θ_2 y θ_3

Para la obtención de θ_2 y θ_3, se puede apreciar mejor si se hace un corte de un plano en Z y la posición que se encuentre el efector final entre X y Y, como se observa en la Figura 2.4.

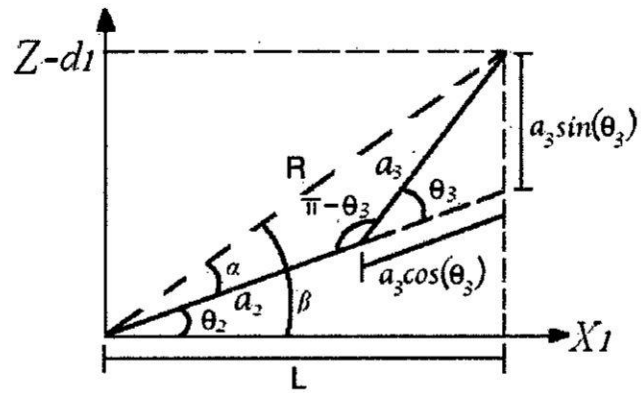


Figura 2.6: Asignación del sistema coordenado para θ_2 y θ_3

De la Figura 2.6 se observa que θ_2 es la resta de β − α donde β es el ángulo formado por los segmentos $Z_c - d_1$ y L; α es el ángulo formado por los segmentos $a_3 \sin\theta_3$ y $a_3 \cos(\theta_3) + a_2$, para determinar θ_2 se utilizarán relaciones trigonométricas obtenidas a partir del triángulo formado por los segmentos de la Figura 2.6.

Para obtener θ2:

$$\beta = \text{atan2}(Z_c - d_1, L) \quad (2.10)$$

$$\alpha = \text{atan2}(a_3\sin(\theta 3), a_3\cos(\theta_3) + a_2) \quad (2.11)$$

$$\theta_2 = \beta - \alpha \quad (2.12)$$

Sustituyendo (2.9) y (2.10) en (2.11), se obtiene que θ2:

$$\theta_2 = atan2(Z_c - d_1) - atan2(a_3 \sin\theta_3, a_3 \cos\theta_3 + a_2) \quad (2.13)$$

Para obtener θ_3 se toma en cuenta el triángulo imaginario formado por a_2 y a_3 de la Figura 2.6:

$$R^2 = a_2^2 + a_3^2 + 2a_2a_3\cos(\pi - \theta_3) \quad (2.14)$$

Despejando $\cos(\pi - \theta_3)$ de (2.13) se tiene:

$$\cos(\pi - \theta_3) = \frac{a_2^2 + a_3^2 - R^2}{2a_2a_3} \quad (2.15)$$

Simplificando (2.14):

$$\cos(\theta_3) = \frac{R^2 - a_2^2 - a_3^2}{2a_2a_3} = D \quad (2.16)$$

De (2.15) se obtiene θ_3 en función a D, tomando en cuenta que se desea controlar con codo arriba (-):

$$\theta_3 = atan2(-\sqrt{1 - D^2}, D) \quad (2.17)$$

2.3 Modelo Dinámico

El modelo dinámico puede ser formulado por medio de cantidades de energía. El modelo de un sistema mecánico rígido, con n grados de libertad, puede ser obtenido usando el método de Euler-Lagrange. Este método proporciona las ecuaciones de estado en forma explícita y estas pueden ser utilizadas para analizar y diseñar estrategias de control. Para implementar estrategias de control es muy importante tener en cuenta todas las fuerzas que interactúan con el robot, como lo son las inercias, fuerzas de Coriolis, centrífugas y el efecto de la gravedad sobre cada uno de los eslabones.

Para modelar el Robot se utilizarán las ecuaciónes de Euler - Lagrange, estas ecuaciones hacen un balance de energías de la dinámica, tanto cinética como potencial, este análisis se puede representar como un lagrangiano de un sistema mecánico dado como:

$$L = K - P$$

Donde K es la energía cinética y P es la energía potencial, por lo tanto la ecuación Euler- Lagrange para cada grado de libertad del manipulador es:

$\frac{d}{dt}\left[\frac{\partial L}{\partial \dot{q}_i}\right] - \frac{\partial L}{\partial q_i} = T_i$

Para i = 1... n

Estas ecuaciones pueden escribirse en forma matricial como:

$$H(q)\ddot{q} + C(q,\dot{q})\dot{q} + D(\dot{q}) + g(q) = T \qquad (2.18)$$

donde la matriz $H(q) \in \mathbb{R}^{nxn}$ es la matriz de inercia del robot, $C(q,\dot{q}) \in \mathbb{R}^{nxn}$ es la matriz de Coriolis, $D \in \mathbb{R}^{nxn}$ es una matriz diagonal con los coeficientes de fricción y $g(q) \in \mathbb{R}^{nxn}$ es el efecto de las fuerzas gravitacionales ejercidas sobre cada articulación.

2.3.1 Momento de inercia del robot de 3 G.D.L.

El momento de inercia depende de la geometría del cuerpo y de la posición del eje de giro. La inercia es la propiedad de los materiales a oponerse al cambio en el movimiento. En el caso más general, la inercia rotacional debe representarse por medio de un conjunto de momentos de inercia y componentes que forman el tensor de inercia. Para el caso de este trabajo los tensores de inercia se consideraron como una varilla rígida, por lo que los tensores de inercia del robot son:

$$I_1 = \begin{bmatrix} l_1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & l_1 \end{bmatrix}$$

$$I_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & l_2 & 0 \\ 0 & 0 & l_2 \end{bmatrix}$$

$$I_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & l_3 & 0 \\ 0 & 0 & l_3 \end{bmatrix}$$

$$I_n = \frac{m_n l_n^2}{12}$$

El cálculo de la matriz de inercia del robot se obtiene mediante la energía cinética que cada articulación genera, ya que dependen únicamente de la masa y de su velocidad. La representación de la Matriz de inercia de un robot para n grados de libertad se representa como:

$$H(q) = \sum_{i=1}^{n} (m_i J_{vci}^T J_{vci} + J_{wci}^T {}^{\circ}R_{ci} I_{ci} {}^{\circ}R_{ci}^T J_{wci}) \qquad (2.19)$$

Dónde:

- m_i es la masa de cada eslabón
- J_{vci} es el Jacobiano de velocidad lineal del eslabón i
- J_{wci} es el Jacobiano de velocidad angular del eslabón i
- ${}^{\circ}R_{ci}$ Matrices de rotación del sistema coordenado i respecto al sistema base
- I_{ci} Tensor de inercia.

Se realizó el análisis necesario para el modelado del robot de 3GDL, tomando como base las matrices de velocidad cinemática del Robot Omni Phantom, por lo que se obtuvieron como resultado:

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \qquad (2.20)$$

De la matriz **H(q),** se definieron los parámetros b_1 , b_2 y b_3 como se observa en la ecuación (2.20), (2.21) y (2.22):

$$b_1 = m_3 O_{c3} + \frac{1}{12} m_3 a_3^2 \qquad (2.21)$$

$$b_2 = a_2 m_3 O_{c3} \qquad (2.22)$$

$$b_3 = m_2 O_{c2} + a_2^2 m_3 + \frac{1}{12} m_2 a_2^2 \qquad (2.23)$$

La matriz H(q) está definida como:

$h_{11} = b_1 c_3^2 + 2b_2 c_2 c_3 + b_3 c_2^2$

$h_{12} = 0$

$h_{13} = 0$

$h_{21} = 0$

$h_{22} = 2b_2 s_2 s_3 + 2b_2 c_2 c_3 + b_1 + \theta_3$

$h_{23} = b_2 s_2 s_3 + b_2 c_2 c_3 + b_1$

$h_{31} = 0$

$h_{32} = b_2 s_2 s_3 + b_2 c_2 c_3 + b_1$

$h_{33} = b_1$

2.3.2 Coriolis y fuerzas centrifugas

La fuerza de Coriolis ocurre cuando un objeto, al desplazarse sobre cualquier sistema rotacional sufre una aceleración adicional producida por una fuerza perpendicular al movimiento. El resultado que provoca al objeto, es una desviación de su recorrido que da lugar a una trayectoria curva.

En cambio las fuerzas centrífugas son aquellas que restringen a un cuerpo a rotar alrededor de un punto, se alejan del centro de masa del movimiento circular uniforme. Para obtener la fuerza de Coriolis y fuerzas centrífugas, se utilizarán los símbolos de Christoffel:

$$C_{kij} = \frac{1}{2}\left(\frac{\partial h_{ij}}{\partial q_k} + \frac{\partial h_{ik}}{\partial q_j} + \frac{\partial h_{kj}}{\partial q_i}\right) \qquad (2.24)$$

La matriz $C(q, \dot{q})$ puede ser fácilmente calculada a partir de la matriz de inercia. La matriz de Coriolisis está dada por:

$$C(q, \dot{q}) = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \qquad (2.25)$$

Donde cada elemento de C(q , $\dot{q}$) es:

$c_{11} = -s_2 \dot{q}_2 (b_3 c_2 + b_2 c_3) - b_1 s_2 c_3 \dot{q}_3 - b_2 s_3 c_2 \dot{q}_3$

$$c_{12} = b_3 s_2 c_2 \dot{q}_1 - b_2 s_2 c_3 \dot{q}_1$$

$$c_{13} = -b_1 s_3 c_3 \dot{q}_1 - b_2 c_2 s_3 \dot{q}_1$$

$$c_{21} = b_3 s_2 c_2 \dot{q}_1 - b_2 s_2 c_3 \dot{q}_1$$

$$c_{22} = b_2 (c_2 s_3 - s_2 c_3)(\dot{q}_2 - \dot{q}_3)$$

$$c_{23} = -b_2 (c_2 s_3 - s_2 c_3)(\dot{q}_2 + \dot{q}_3)$$

$$c_{31} = b_1 s_3 c_3 \dot{q}_1 - b_2 c_2 s_3 \dot{q}_1$$

$$c_{32} = 2 b_2 (c_2 s_3 - s_2 c_3)\dot{q}_2$$

$$c_{33} = 0$$

2.3.3 Coeficientes de fricción

Para obtener un mejor análisis se deben de tomar en cuenta las fuerzas que se oponen al movimiento de las articulaciones. En los sistemas mecánicos la fricción viscosa es un elemento que permite que un objeto se mueva con mayor amortiguamiento. En la representación matricial para el modelo, los coeficientes serán los elementos de la diagonal principal de la matriz D.

$$D = \begin{bmatrix} C_{f1} & 0 & 0 \\ 0 & C_{f2} & 0 \\ 0 & 0 & C_{f3} \end{bmatrix} \tag{2.26}$$

Los coeficientes de fricción se puede obtener de la hoja de datos de los motores, si el fabricante no la específica, este se puede calcular como:

$$CF = \frac{Max.Par[Nm]}{Max.Velocidad[rad/seg]}$$

2.3.4 Fuerzas gravitacionales del robot

Las fuerzas gravitacionales del robot se obtienen del efecto generado por la energía potencial. Para conocer el vector g(q), se obtiene primero el vector de constantes gravitacionales y la energía potencial de cada eslabón del robot.

$$g = [0 \quad 0 \quad 9.8]\left[\frac{m}{s^2}\right] \tag{2.27}$$

La Energía Potencial para n eslabones se representa como:

$$M = \sum_{i=0} m_i g^T r_{ci} \tag{2.28}$$

Donde m_i es la masa de cada eslabón, g es un vector con la constantes gravitacional ($9.81\frac{m}{s^2}$) en su componente en z_0, referido al sistema base y r_{ci} es el vector posición para cada instante de cada centro de masa respecto al sistema base. Del resultado de la energía potencial del robot se obtiene el vector de fuerzas gravitacionales dado por:

$$g(q) = \begin{bmatrix} \frac{\partial P}{\partial q_1} \\ \frac{\partial P}{\partial q_2} \\ \frac{\partial P}{\partial q_3} \end{bmatrix} = \begin{bmatrix} 0 \\ (m_2 l_{c2} + a_2 m_3) g c_2 \\ m_3 l_{c3} g c_3 \end{bmatrix} \tag{2.29}$$

2.4 Control PID para el Robot

En la actualidad existen varios métodos de control conocidos, sin embargo, en la industria los controladores PD y PID son los que suelen ser ocupados, debido a que tienen un gran desempeño y facilidad de implementación en los distintos procesos industriales.

Para comprobar el análisis de la cinemática directa, cinemática inversa y el modelo dinámico del Robot se utilizó un controlador PID, debido a su fácil implementación y no se necesario incorporar parte del modelo dinámico en el controlador.

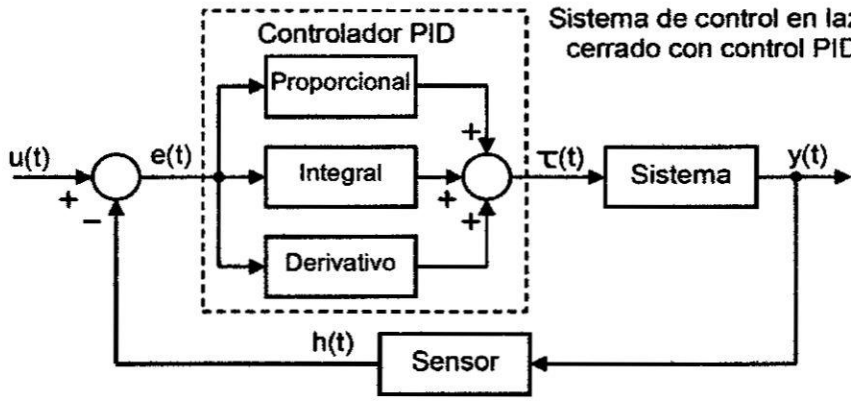
$$\tau = K_p \Delta q + K_d \frac{d}{dt} \Delta q + K_i \int_0^t \Delta q dt + g(q) \tag{2.30}$$

Las ganancias del controlador PID con compensación de gravedad son K_p, K_i, K_d y g(q) como se muestra en la ecuación (2.30).

- La ganancia K_p, representa un cambio presente en la salida del controlador, permite generar una compensación en la medición.
- La ganancia K_i, da una respuesta proporcional a la integral del error. Esta acción elimina el error en estado estacionario, provocado por la acción proporcional. Su implementación da como resultado un mayor tiempo de establecimiento, una respuesta más lenta y el periodo de oscilación es mayor que en el caso de la acción proporcional.
- La ganancia K_d, da una respuesta proporcional a la derivada del error (velocidad de cambio del error). Añadiendo esta acción de control a las anteriores se disminuye el exceso de sobreoscilaciones.
- La compensación de gravedad g(q), se define en la ecuación (2.29),
- depende de las posiciones articulares del robot.

Figura 2.7: Diagrama de bloques de un sistema en lazo cerrado

Figura 2.7: Diagrama de bloques de un sistema en lazo cerrado



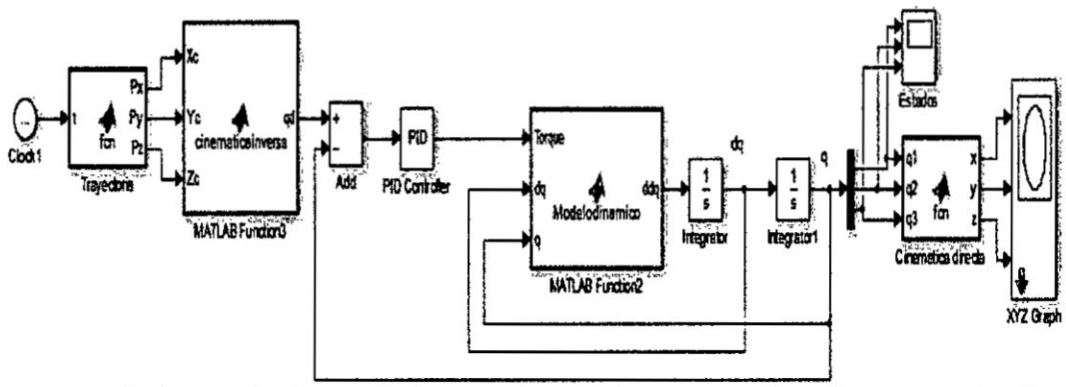
En la Figura 2.7 se ve diagrama de bloques del sistema en lazo cerrado con un controlador PID, donde el bloque de sistema representa el modelo dinámico del Robot, en la simulación la salida del modelo dinámico representa los datos censados. Para el caso de este trabajo la entrada u(t)

es representada como las posiciones articulares deseadas y y(t) representa la salida del sistema.

2.4.1 Generación de trayectorias

La generación de trayectorias es uno de los aspectos básicos del desarrollo de robots industriales. Permite al robot poder desplazarse de un lugar a otro de manera segura, a partir de un modelo de obstáculos que lo rodean y a un camino, ya calculado. Los estudios en generación de trayectorias son importantes debido a que son la base del desplazamiento de un robot. Para esto, en primer lugar, se debe de calcular una trayectoria. Ésta puede ser calculada por distintos métodos dependiendo del algoritmo utilizado. Una vez calculada la trayectoria, debe ser realizada por el robot real, lo que lleva a un problema de incertidumbre en su ejecución.

Es recomendable simular el comportamiento del robot antes de probarlo en uno real, debido a que se pueden generar fallas en el control. En la simulación se busca obtener un comporta- miento ideal del robot, ya que



un seguimiento de trayectoria exacto se logra cuando el error entre las trayectorias de referencia y las reales es cero. En un robot real esto no suele pasar y lo que se busca es hacer este error tan cercano a cero como sea posible, tomando como base los datos de la simulación.

Figura 2.8: Modelo del robot para el seguimiento de trayectoria

En la Figura 2.8 se observa el diagrama de bloques utilizado en SIMULINK que representa el sistema en lazo cerrado del controlador y el modelo dinámico del Robot. Para el robot se tomaron condiciones iniciales de las

posiciones articulares como q = [0; 0.5; −1.2] y se consideró que cuando el tiempo es cero la velocidad en cada una de las articulaciones es cero.

2.4.2 Seguimiento de trayectoria circular

En el diseño de trayectorias se optó por evaluar el robot con una trayectoria circular. Este tipo de trayectoria permite describir un comportamiento sencillo del robot, cuando el robot termina de generar la circunferencia se repite la misma trayectoria en el espacio de trabajo hasta que concluya el tiempo de simulación. Para poder generar una trayectoria circular se programan las ecuaciones paramétricas de una circunferencia, para el caso de este trabajo la trayectoria se diseñó para el plano XY, y para el eje Z se realiza un corte en una altura que el robot pueda alcanzar.

En las ecuaciones paramétricas se debe especificar las proporciones que debe tomar la trayectoria, ya que si se hace muy grande el robot no alcanzará el punto deseado y puede que se generen singularidades. Para este trabajo las ecuaciones paramétricas de la circunferencia son representadas en el plano XY y el eje Z.

Para el plano XY se consideraron las siguientes ecuaciones:
Donde,

$x = 19.5 + r\cos(\omega t)\,[cm]$ (2.31)

$Y = 19.5 + r\sin(\omega t)\,[cm]$ (2.32)

r: representa el radio de la circunferencia, para este trabajo se tomó un radio de 15 [cm].

ω: es la frecuencia a la que el robot se desplaza en la trayectoria, para que el robot lo realice a una velocidad regular se tomó como 0.6[*rad/seg*].

Para el eje Z se hizo un corte en:

$$z = 10[cm] \qquad (2.33)$$

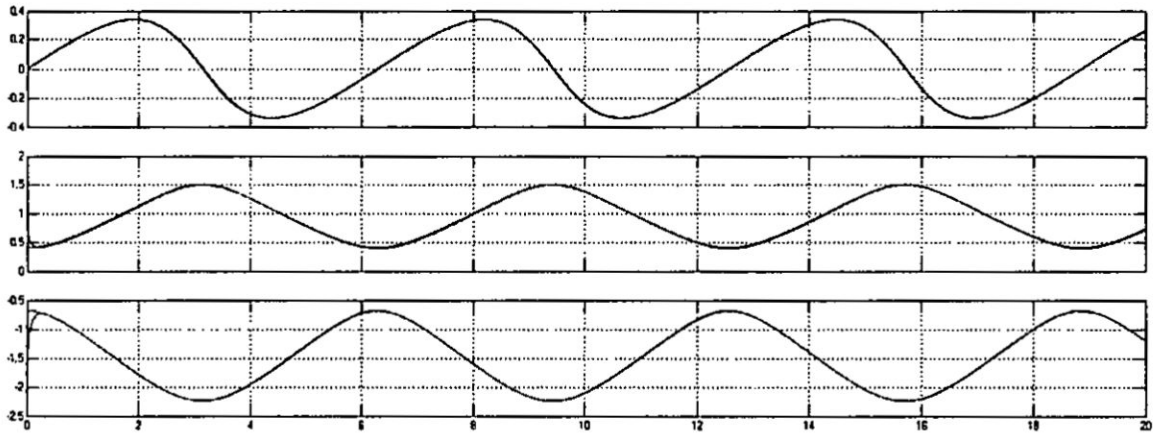


Figura 2.9: Comportamiento de los ejes del Robot

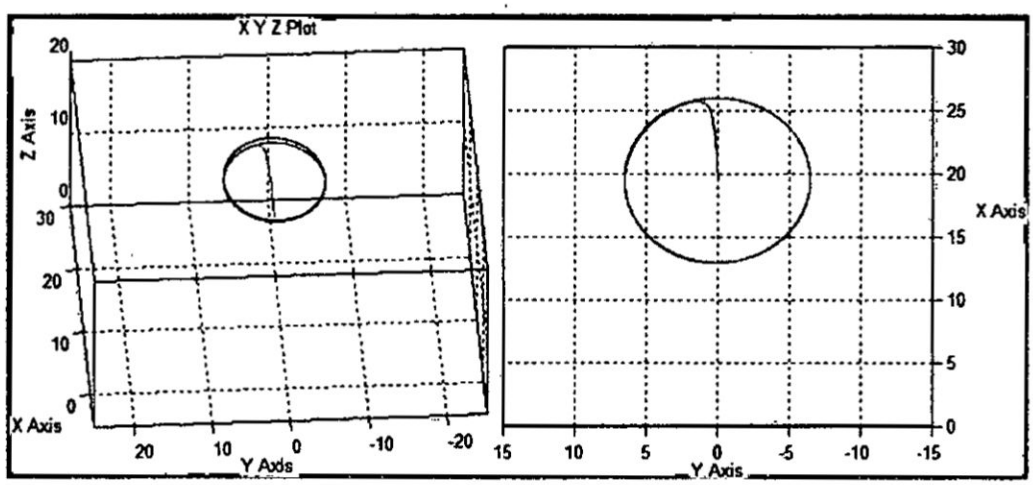


Figura 2.10: Grafica de la trayectoria en el sistema coordenado

De las Figuras 2.9 y 2.10 se pueden observar de color azul la trayectoria que debe de seguir el efector final del robot y la de color verde es la que generó el robot con respecto al análisis realizado de las cinemáticas y dinámicas del robot.

2.4.3 Seguimiento de trayectoria mediante parametrización

Existen distintas formas de generar trayectorias, como se observó en el seguimiento de trayectoria circular, es necesario enviarle al robot posiciones deseadas mediante ecuaciones que representan la trayectoria,

también se puede hacer uso del efector final durante la ejecución de la trayectoria. Para el caso de este seguimiento de trayectoria se desea trasladar un objeto desde un punto del espacio de trabajo a otro punto.

En el seguimiento circular solo se ingresa una ecuación para cada eje coordenado, pero es posible enviar una trayectoria mediante la parametrización de ecuaciones como se muestra a continuación.

En la trayectoria del Robot se definen las siguientes ecuaciones:

- r=33 cm, r representa una distancia máxima a la que el brazo llegará, este es similar al del seguimiento de trayectoria circular.
- w=0.5, w representa la velocidad a la que va realizar la trayectoria.
- t: representa el tiempo real.
- $t_1 = t - 1$, en esta ecuación se hace una diferencia para reducir el tiempo un segundo.
- $t_2 = w * t_1$, este es un múltiplo del tiempo menos uno.
- Pinza=0, el valor de la pinza varía en donde 0 se encuentra totalmente abierta y cuando está en 100 se cierra completamente.

En el Robot se tomarón condiciones iniciales igual a x=20, y=0, z=2 Las condiciones iniciales permanecerán mientras el tiempo *t2* sea menor a cero. En este caso el Robot ya sujeto la pieza a trasladar.
Cuando el tiempo t_2 sea mayor a cero y menor a 3.1416, empezara a generar un semicírculo en el plano XY.

$$x = r\cos t_2$$
$$y = r\sin t_2$$

Para el eje Z si el tiempo es mayor a cero y menor a 1.57 empezará a alzar el brazo hasta llegar al tiempo t_2=1.57.
z = 8.912655 $*$ (t_2) + 2

Al llegar al tiempo t2 el efector final alcanza una altura de 16[cm] con respecto a Z0.

Para el eje Z si el tiempo es mayor a 1.57 y menor a 3.1416 empezará a bajar el brazo hasta llegar al tiempo $t_2 = 3.1416$

$z = -8.912655 * (t_2) + 30$

Al finalizar el brazo deberá permanecer estable en la posición x=0, y=−20, z=2.

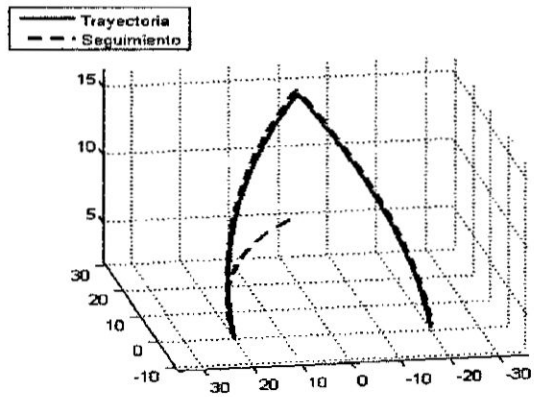


Figura 2.11: Grafica del seguimiento de trayectoria mediante parametrización en el sistema coordenado

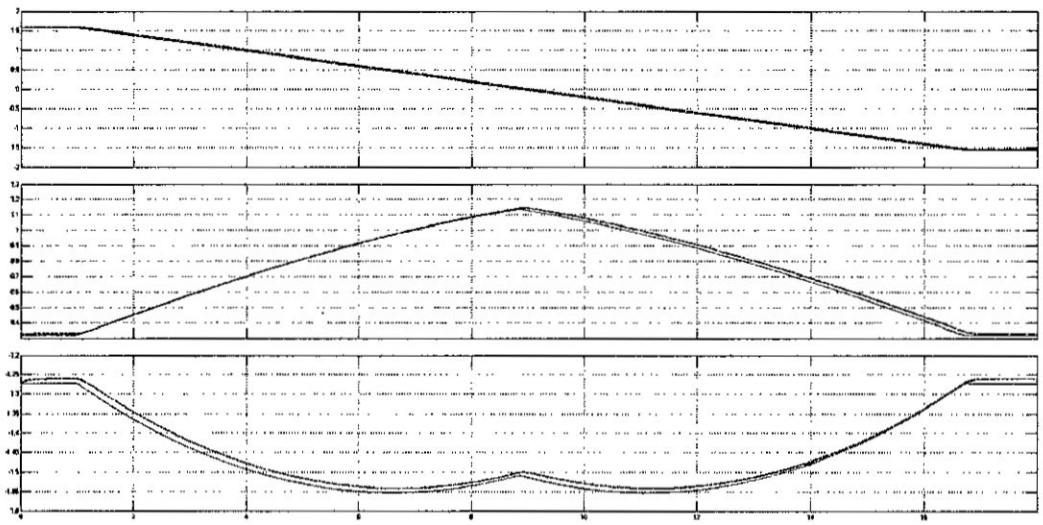


Figura 2.12: Graficas de las articulaciones durante seguimiento de trayectoria mediante parametrización

En la Figura 2.11 se observa con línea punteada la trayectoria generada por el Robot en el sistema coordenado, en él se ve que toma las condiciones iniciales del sistema coordenado hasta llegar a las condiciones finales. También se puede notar que la trayectoria en el eje Z va mostrando una pendiente que empieza a incrementar y al llegar a la mitad de la trayectoria decrece, la trayectoria en Z forma un triángulo, es preferible que se generen trayectorias más suaves. Si se desea realizar una trayectoria más suave se dificulta un poco más al obtener las ecuaciones de la trayectoria

En la Figura 2.12 se observa el seguimiento de trayectoria visto desde cada posición articular, donde la trayectoria punteada es la trayectoria que generada por el robot.

Se puede ver que existen distintas formas de generar trayectorias, algunas se calculan con métodos numéricos, debido a que le dan un mejor comportamiento al robot.

III. Diseño del Robot de 3 GDL modelado en esta Tesis.

Tanto la estructura mecánica como lógica de un robot, son parte fundamental en la construcción del mismo. En este capítulo se detallaran los diseños en hardware (mecánico) que se tomaron en base a un modelo de robot para fines educativos al cual al final se le acoplara una pinza en el extremo para fines didácticos, las adaptaciones en software para dicho modelo de Robot de 3G.D.L si se tuvieron que acondicionar por nuestra cuenta con la finalidad de comunicar a los servomotores con la interfaz en MATLAB - SIMULINK.

Cave recalcar que este modelo ya fue diseñado y solo estamos adjuntando los parámetros y procedimientos de fabricación que incluyo su diseñador, por ser todo estos parámetros de vital importancia para establecer la sintonización adecuada en el controlador que se incluirá en el entorno SIMULINK MATLAB para establecer el seguimiento de trayectorias .

3.1. COMPONENTES DEL ROBOT

En el diseño de un robot se debe conocer el material necesario para poder realizar una tarea específica. Para el diseño se debe tomar en cuenta dos partes principales: el diseño físico (hardware) y el diseño lógico (software).

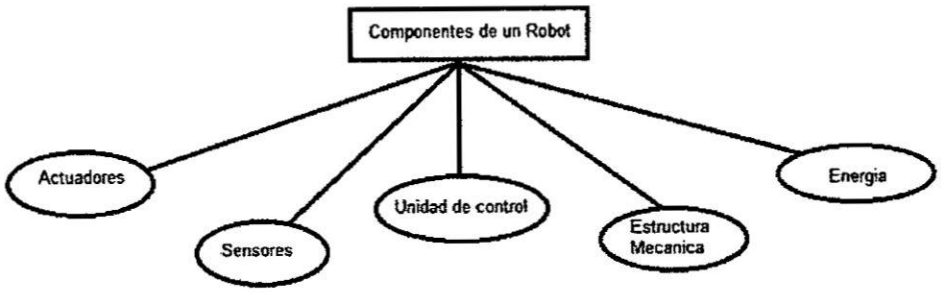


Figura 3.1: Diagrama de componentes

En la Figura 3.1 se observan todos los componentes que conforman el Robot, en donde algunos elementos del robot están destinados a obtener información (sensores y comunicación), otros al procesamiento (control) y otros más generan respuestas al medio (estructura mecánica y actuadores). El diseño físico está compuesto por los actuadores, sensores y la estructura mecánica, y el diseño lógico está compuesto de toda la comunicación del Robot y el control del mismo.

3.1.1 Estructura mecánica

La estructura mecánica es el esqueleto del diseño mecánico, y da soporte a todos los elementos y define los movimientos que el robot podrá realizar. Por ello para que este prototipo se pueda implementar se define el número de grados de libertad (GDL) que contendrá el brazo robótico, así como la distribución y orientación de cada uno de ellos a lo largo de la estructura mecánica.

El primer aspecto es la cantidad de grados de libertad del robot. Mientras más grande sea la cantidad de grados de libertad, aumenta la cantidad de

movimientos que el mismo pueda realizar, generando redundancias, lo que provoca una mayor complejidad en el modelo dinámico del robot.

Como se ha mencionado con anterioridad que el robot a modelar es uno de 3 grados de libertad como se muestra en la Figura 3.2.

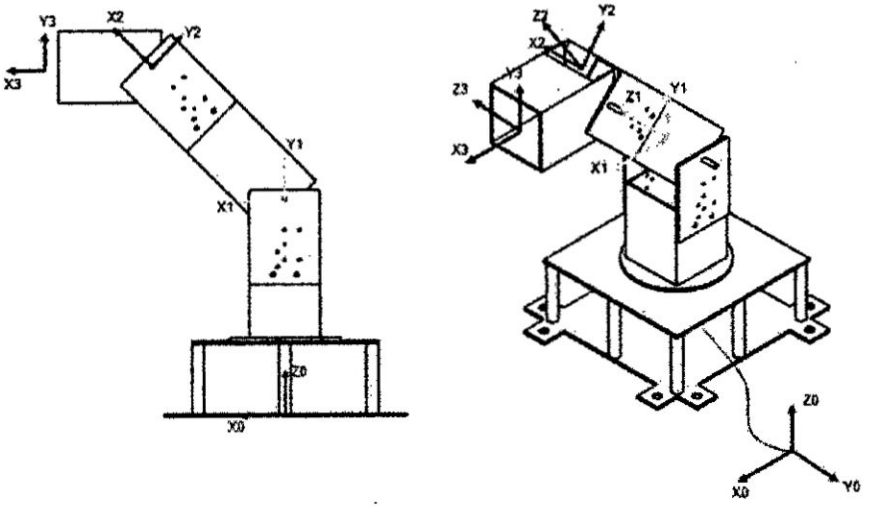


Figura 3.2: Modelo del Robot de 3 GDL

Debido a que el robot desarrollado en este trabajo no contiene muñeca esférica, no permite que al efector final se le pueda asignar alguna orientación deseada.

Mecánicamente, un robot está formado por una cadena cinemática abierta o cerrada dependiendo del tipo de configuración del mismo. La constitución física de la mayor parte de los robots industriales guarda ciertos rasgos antropomórficos, por lo que en ocasiones, para hacer referencia a los distintos elementos que componen el robot, se usan términos como cuerpo, brazo, codo y muñeca.

En este trabajo la representación de los eslabones del robot son el brazo y antebrazo. Se modelan articulaciones de tipo rotacional, ya que son las utilizadas en el arreglo cinemático del robot antropomórfico.

3.1.2 Actuadores y Sensores

Los actuadores son los que transforman las señales provenientes del controlador en movimiento de las articulaciones. Por lo tanto, son los encargados del movimiento del manipulador. Los accionadores utilizados en la robótica de manipuladores son básicamente los utilizados en otro tipo de máquinas. Pero son tres los tipos de accionadores más utilizados para generar el movimiento. Los tipos de actuadores se pueden clasificar según el tipo de energía que utilizan, por lo tanto se consiguen del tipo eléctrico, neumático e hidráulico. El tipo de actuador a utilizar siguiendo los lineamientos del diseñador en este robot modelado es del tipo eléctrico, ya que la función principal será de tipo educativa, y por consiguiente no requiere del manejo de cargas pesadas.

De estos tipos de actuadores existen tres tipos:

- **Motores eléctricos de corriente continua.** Estos se utilizan para proporcionar movimientos giratorios en los que no se requiere mucha precisión.

- ***Motores paso a paso.*** Estos permiten controlar de forma precisa el ángulo de giro del motor, haciendo que el motor se coloque en una posición determinada. Para el control de estos motores se requiere un circuito electrónico de control.

- ***Servomotor de Modelismo.*** Conocido generalmente como servo o servo de Modelismo, Es un motor de corriente continua que tiene la capacidad de ser controlado a alguna posición deseada. Es capaz de ubicarse en cualquier posición dentro de un rango de operación (generalmente de 180°) y mantenerse estable en dicha posición. Los servomotores son muy utilizados en robots de pequeña escala, por la característica que integran

tanto el motor como la electrónica de control y la mecánica de reducción en un solo dispositivo.

- **Estructura de un servomotor**

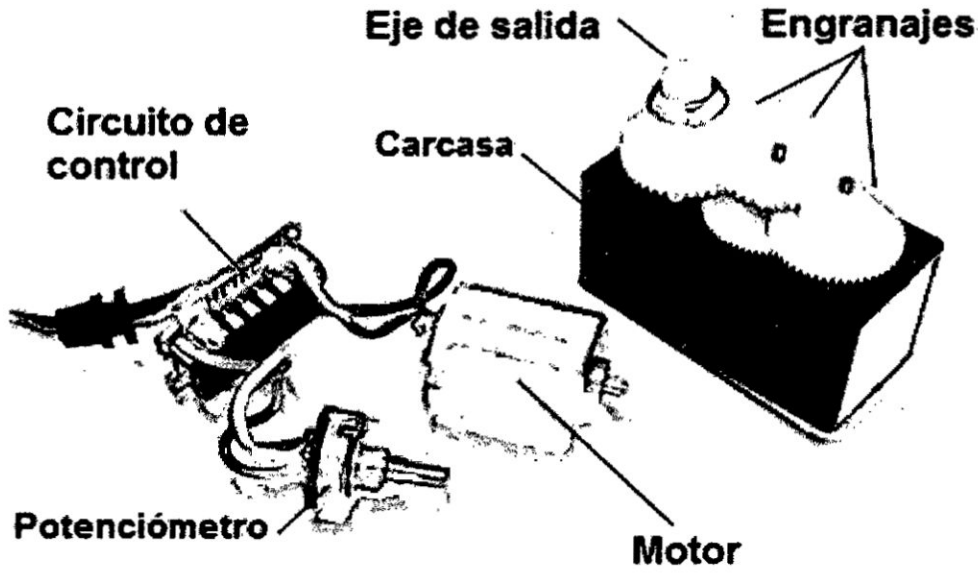


Figura 3.3 estructura básica de un servomotor

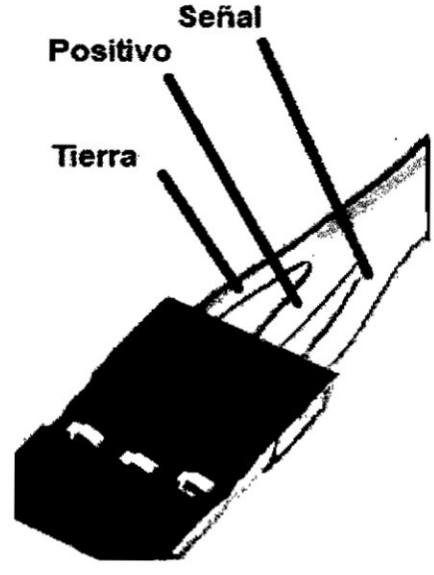


Figura 3.4 cable de 3 polos del servomotor

Como se puede observar en las figuras 3.3 y 3.4, un servomotor está compuesto básicamente por un motor que tiene un potenciómetro conectado a su eje, todo ello gobernado por un circuito de control. El eje del motor se encuentra, a su vez, conectado a un tren de engranajes que reduce la velocidad del motor y la convierte en fuerza en el eje de salida.

Todo ello está contenido en una carcasa.

Al circuito de control llega un cable que, por lo general, tiene 3 polos: con dos se fija la tensión que llegará al motor: tierra y positivo (normalmente, entorno a los 6 V), y con el tercero, se envía la señal con la que se logrará indicar, como se explicará más adelante, al servo el giro que ha de hacer y la fuerza a proporcionar.

Por otro lado los sensores, son los encargados de recoger la información del entorno y enviarla a la unidad de control para su procesamiento. Los sensores se pueden clasificar en dos tipos dependiendo de la función que realicen.

- ***Sensores externos.*** Toman datos del entorno (como sensor de voltaje que mide la fuente de alimentación, para ver si no supera el rango de operación de los motores y sensor de temperatura, para evitar el sobrecalentamiento).

- ***Sensores internos.*** Controlan el propio funcionamiento del robot (como sensor de posición y sensor de velocidad).

Los actuadores electromecánicos se utilizan principalmente en robots con articulaciones rotacionales. En los robots es muy común el uso de los servomotores como es el caso de este trabajo, debido a que tienen una facilidad de integración.

3.1.3 Servomotor AX-12

Para las articulaciones del robot se utilizaron los servomotores Dynamixel AX-12A de RO- BOTIS, cuyas principales características se pueden observar en la hoja de datos del servomotor. El servomotor Dynamixel AX-

12A puede ser llevado a posiciones angulares específicas al enviar una señal codificada. Mientras el servomotor reciba la misma señal codificada mantendrá la posición angular del piñón hasta que la señal codificada sea distinta, la posición angular del piñón cambiará.

Los servomotores Dynamixel pueden llegar a ser mejores que algunos servomotores de RC. Los servomotores Dynamixel tienen internamente un microcontrolador, el cual adquiere la información necesaria de los sensores y permite que el usuario mediante la comunicación serial pueda configurar algunos parámetros como el par, la posición, etc. En cambio los servomotores de RC internamente tienen una serie de circuitos integrados que le permiten adquirir los datos y envía o recibe datos mediante una señal PWM.

Figura 3.5: Servomotor AX-12

El servomotor Dynamixel AX-12A está compuesto de una serie de sensores, una memoria EEPROM y una memoria RAM interconectados con el microcontrolador. Está integrado de un potenciómetro encargado de medir la posición del servomotor con respecto al voltaje, entre otros se encuentra un sensor de temperatura que permite que el servomotor no se queme por sobrecalentamiento, un sensor de voltaje que se encarga de medir la tensión del mismo para que no haya un sobrepaso. Si el sensor de voltaje y el sensor de temperatura tienen un cambio que pueda afectar al

servomotor, el microcontrolador bloquea el funcionamiento del motor hasta que se corrija el error.

ROBOTIS en su diseño de los servomotores ocupan un microcontrolador Atmega8 como se observa en la Figura 3.6, en el es fundamental el uso de una memoria EEPROM y una memoria RAM. Las memorias EEPROM y RAM se encargan de enviar y almacenar los datos del motor para ser transferidos al microcontrolador después de que la alimentación sea conectada o desconectada respectivamente.

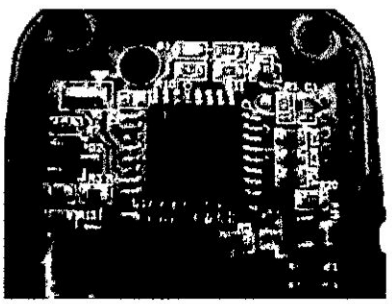


Figura 3.6: Microcontrolador Atmega8 del servomotor AX-12

El fabricante maneja una gama de servomotores que tienen algunas diferencias entre sí, como es el par que puede soportar uno con respecto al otro, el tamaño, peso, precio, tipo de conector, etc. El servomotor Dynamixel AX-12A es uno de los de gama más baja por lo que soporta 1.5 [Nm] de par con una alimentación de voltaje que se recomienda de 11 V.

Cada servomotor tiene 2 conectores, donde cada conector tiene 3 canales como se muestra en la Figura 3.7. El primer canal representa la tierra o GND, el segundo canal es la alimentación de voltaje de aproximadamente 11V, el tercer canal representa la entrada o salida de datos; el otro conector se utiliza cuando se desea utilizar más de un solo servomotor como es el caso de este trabajo. El tipo de conector depende del servomotor que se utilizará, debido a que los que son de gama mayor suelen tener un conector con 4 puertos, estos conectores son diseñados por el propio fabricante.

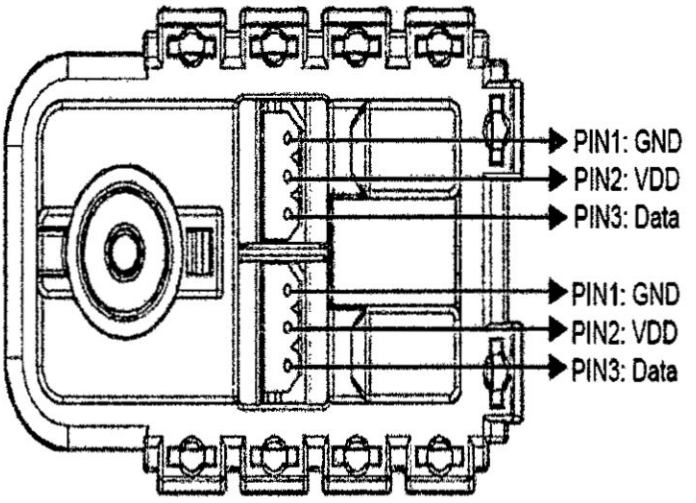


Figura 3.7: Diagrama de Conexión del servomotor Dynamixel AX-12

El servomotor AX-12 tiene la facilidad de modificar algunos parámetros del mismo, por lo que se utiliza un programa en específico conocido como RoboPlus. Para la configuración del servomotor es necesario tener un dispositivo de ROBOTIS conocido como USB2Dynamixel, ya que tiene compatibilidad con el software RoboPlus.

La USB2Dynamixel es un dispositivo que se utiliza para operar los servomotores Dynamixel desde un ordenador, mediante la comunicación serial. Este se conecta al puerto USB del ordenador y a los conectores 3P y 4P que están asignados a los servomotores. El uso de la USB2Dynamixel es restringido, esto se debe a que tiene sus propias librerías que son compatibles con MATLAB, LabView, Visual Studio C++ y Visual Studio C#.

Para poder realizar un buen diseño del robot se debe de saber las ventajas y desventajas que tiene el uso de los servomotores AX-12A, como se observa en la Tabla 3.1.

Ventajas	Desventajas
El servomotor AX-12 es el más liviano, pequeño y barato del fabricante	El servomotor AX-12 es el que genera menos par
No es necesario modificar algún mecanismo in- terno para obtener las lecturas de los sensores	Se requiere mandar una cadena de bytes para que el servomotor realice una instrucción
Su configuración permite conectar varios servo- motores entre sí(como máximo se pueden conectar los 254 servomotores)	Para obtener respuesta de un sensor, se envía una instrucción al servomotor, recibiendo una cadena de bytes por lo que los últimos 2 bytes representan el valor deseado.
Tiene un indicador, cuando se activa bloquea el funcionamiento del servomotor, debido a que ocurrió un error de comunicación o problemas en alguna parte del servomotor.	El servomotor utiliza un potenciómetro como sensor de posición, la señal medida del potenciómetro es más ruidosa comparada con el codificador incremental.
Existe la posibilidad de mover varios servomotores al mismo tiempo sin recibir una respuesta de ellos.	El fabricante no da mucha información para realizar código abierto sobre los
El cableado es muy sencillo: ocupa 2 cables de alimentación y uno de comunicación UART	
Tiene una resolución de 0.3 grados, puede moverse de una posición 0 hasta 300 grados.	
No necesita una etapa de potencia para alimentar al servomotor	

Tabla 3.1 Ventajas y Desventajas del Servomotor AX-12A

Los servomotores se pueden configurar en modo Articulación o en modo Rueda como se explica en la Tabla 3.2

Modo de Articulación	Modo Rueda
Mueve el eje del motor hasta llegar a un ángulo deseado y velocidad deseada	Existe la posibilidad de girar el eje del motor a un cierto par, dándole el sentido de giro
No es necesario obtener datos de la posición	Se puede adquirir datos en tiempo real de la posición del eje del motor o de algún sensor interno del servomotor
Realiza un control en lazo abierto	Se puede realizar control en lazo cerrado
Actuá de 0° a 300°, donde toma como resolución de 0 a 1023	El potenciómetro va de 0° a 300° y los 60 grados restantes son punto muerto de la lectura de datos por lo que suele representarlo como 0°
Tiene un par que puede variar de 0 a 1.5 Nm, donde toma como resolución de 0 a 1023(el fabricante lo tomá como velocidad, pero en si es una relación con el par de giro del motor)	El par del motor en sentido anti-horario va de 0 a 1.5 Nm, tomando como resolución de 0 a 1023 cuentas. El par del motor en sentido horario va de 0 a 1.5Nm tomando como resolución de 1024 a 2047, en donde 1024 representa el giro en cero y 2047 el máximo giro

Tabla 3.2 Configuración en el giro del motor

En la Tabla 3.2 se muestra una breve explicación de los modos de configuración del motor para poder realizar control en lazo abierto y cerrado que es lo que se desea realizar en este trabajo de tesis.

3.1.4 Unidad de control

La unidad de control es el encargado de analizar la información que les mandan los sensores, tomar decisiones y dar órdenes para que las realicen los actuadores.

La unidad de control se puede representar de dos formas:

- ***Mediante un circuito electrónico que puede ser programable.*** Este sistema de control permite construir pequeños robots sin necesidad de cables de conexión con un ordenador.
- ***Mediante ordenador.*** Este es más utilizado en máquinas que no realizan desplazamientos, ya que la conexión por cable con el ordenador dificultaría su movilidad.

Para el desarrollo de la unidad de control se implementó con un microcontrolador Atmega2560 de Atmel, el cual tiene compatibilidad con el software Arduino. El uso de este microcontrolador provee un compromiso entre flexibilidad y bajo costo, en este se programa gran parte del funcionamiento del hardware, mediante el lenguaje de programación en C. El microcontrolador Atmega2560 por sí solo no se puede comunicar con los motores, por lo que se necesita adicionar un circuito integrado 74LS241, este se encarga de enviar y recibir la información necesaria por un mismo canal. El tipo protocolo que maneja es Half duplex asíncrono de comunicación serie, y el tipo de cableado que ocupa es mediante el cableado Daisy Chain.

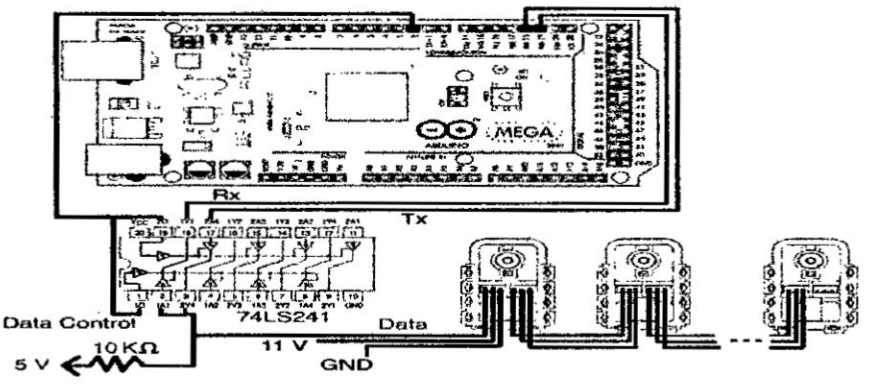


Figura 3.8: Microcontrolador con circuito integrado 74LS241

No es necesario usar más de un circuito integrado 74LS241, ya que los motores tienen la disponibilidad de conectarse entre sí, mediante el tipo de cableado Daisy Chain como se muestra en la Figura 3.8.

Los servomotores para recibir información del microcontrolador, es necesario enviarle una cadena de datos específica para el tipo de instrucción que se desea realizar, en este trabajo solo se usaron instrucciones para la lectura de posición y escritura de par. Para poder realizar algún tipo de instrucción, se debe verificar la tabla de control del Servomotor AX-12 localizada en los manuales del Servomotor. Las instrucciones que realiza el microcontrolador se pueden consultar en el "Apéndice B. Código de Instrucciones del microcontrolador" y si se desea más información de cómo se generan las instrucciones se puede observar en "Apéndice D. Ejemplos de Instrucciones del Servomotor Dynamixel AX 12 y AX 18".

3.1.5 Suministro de energía

Los robots son máquinas electromecánicas que necesitan de un suministro de energía eléctrica que alimenta la unidad de control y los motores eléctricos.

En este trabajo los componentes principales que requieren una fuente de alimentación son los motores eléctricos que representan cada articulación

del robot, la unidad de control y los circuitos integrados que contenga el robot. Para la selección de la fuente de alimentación se debe verificar cuanto consume cada componente, tomando en cuenta que cada motor puede consumir como máximo 0.9 A, la unidad de control y los circuitos integrados consumen aproximadamente 0.3 A. Para fines de este trabajo se utiliza un adaptador de voltaje de 12 V a 3A, mostrado en la Figura 3.8.

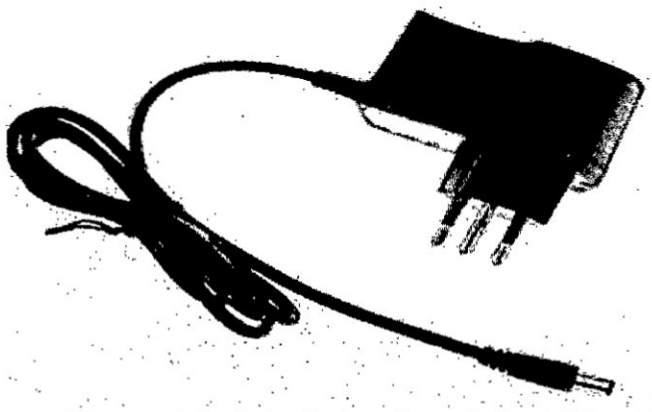


Figura 3.9: Adaptador de voltaje 12V a 3A

3.2. DISEÑO ELECTRÓNICO

Para el diseño electrónico del robot se utiliza una herramienta de diseño de circuitos impresos (Printed Circuit Board, PCB), esto facilita la creación de placas electrónicas para poder tener una mejor comunicación y fuente de alimentación entre los dispositivos. En este trabajo se utilizó Proteus v.8 ISIS/ARES, debido a que es fácil de desarrollar circuitos eléctricos.

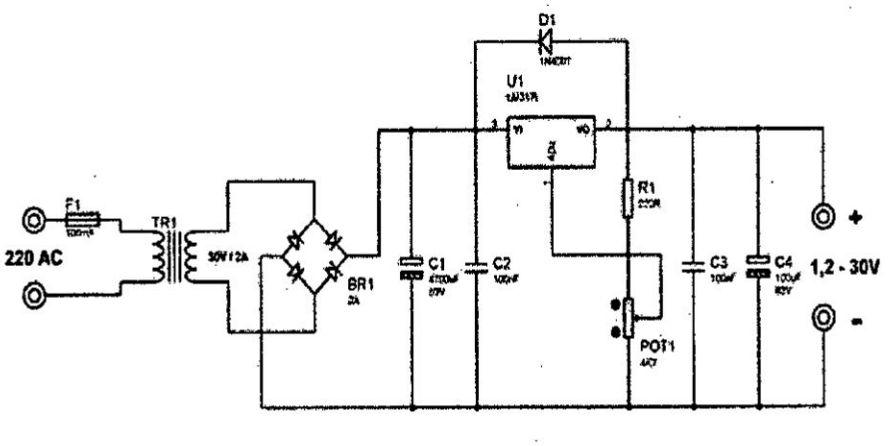


Figura 3.10: Circuito Regulador de Voltaje a 11V

En la Figura 3.10 se muestra este circuito ya que el fabricante recomienda usar los motores a 11V, por lo que se usa una fuente de alimentación regulable en el intervalo de 1,2 a 30 Vcd, esto permite regular el voltaje, los motores se pueden utilizar desde 9V hasta 14V.

3.3. DISEÑO MECÁNICO.

Para el diseño mecánico del robot el autor se emplea una herramienta de diseño asistido por computadora (Computer Aided Design-CAD), ya que facilita el modelado de componentes con las medidas correspondientes, permitiendo un ahorro en recursos, tanto económicos como en tiempo. En la actualidad existe una gran variedad de herramientas CAD, las cuales se pueden elegir dependiendo de las necesidades y los recursos disponibles. En este caso utilizó AutoCAD 2015 debido a que la versión de estudiante está disponible para varios sistemas operativos, lo que lo convierte en una herramienta accesible.

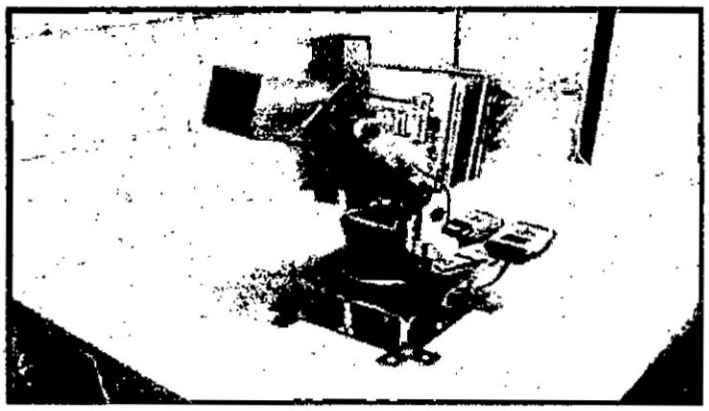


Figura 3.11. Prototipo de robot educativo

Las partes del robot se consideraron en base al tamaño del servomotor Dynamixel AX-12, tratando de copiar la morfología de un brazo humano. Por lo que se definieron algunas distancias para cada parte del mismo.

La cadena cinemática de un robot está compuesta de los eslabones y articulaciones que hay en el mismo. En este trabajo los eslabones del robot son representados como el brazo y el antebrazo, y las articulaciones representan cada uno de los servomotores del robot que son conocidos como la cintura, el hombro y el codo.

Para cumplir con el requisito de poco peso, se trabaja con aluminio como material principal del cuerpo del robot manipulador.

Para simplificar el proceso de fabricación y diseño [autor], se emplea un perfil comercial cuadrado de aluminio de 76,2mm (3 pulgadas) de lado y 2 mm de espesor. Luego a los eslabones 1 y 2 se les colocan extensiones hechas de chapa de aluminio de 3 mm de espesor, con el fin de crear los puntos de unión entre los eslabones. A los eslabones 2 y 3 se les recorta un pedazo de pared para permitir un rango más amplio de movimiento, ver Figura 3.13.

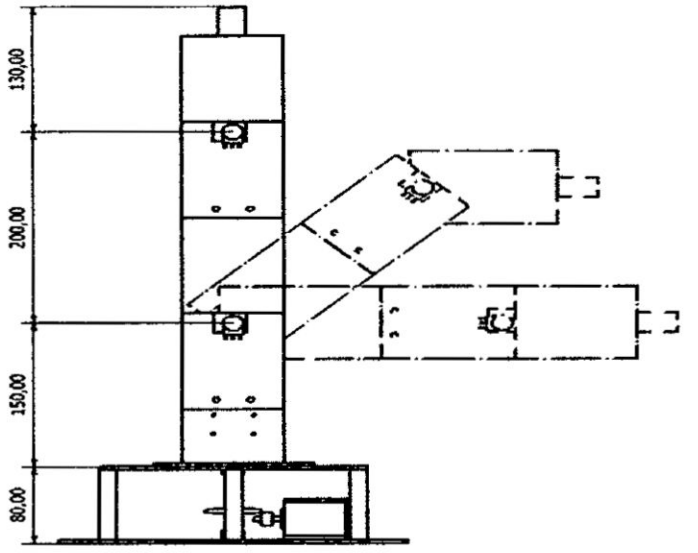


Figura 3.12. Esquema del robot de 3G.D.L. a modelar

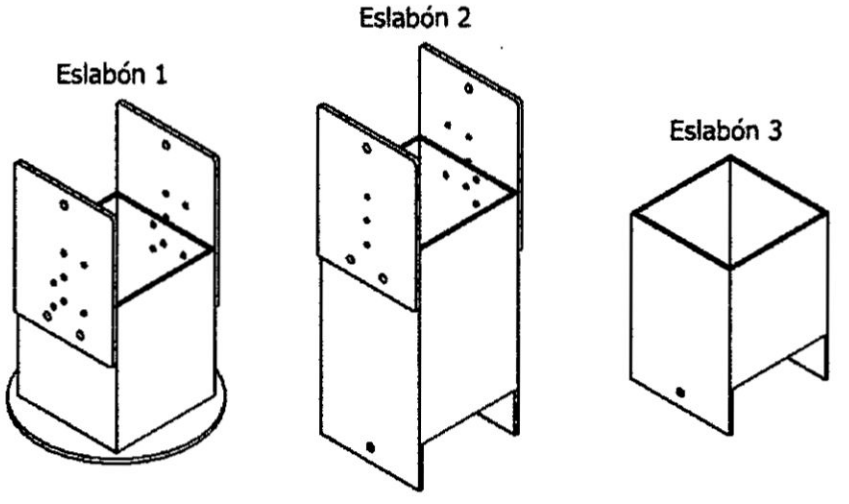


Figura 3.13 Eslabones

La unión entre los eslabones 1 y 2 se realiza por medio de un eje de 5 mm de diámetro, 120 mm de longitud y de material aluminio. El eje se inserta en los agujeros de la parte inferior del eslabón 2, y en la parte superior del eslabón 1, luego se suelda el eje al eslabón 2. De igual manera para la unión del eslabón 2 y el eslabón 3, en este caso el eje va soldado al eslabón 3. Ver Fig. 3.13 unión de eslabones.

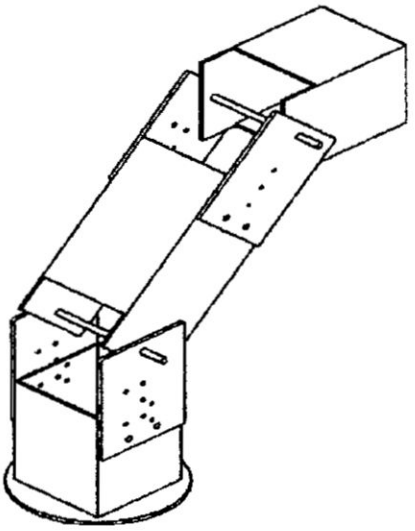


Figura 3.14. Unión de eslabones

Luego se construye la base del manipulador, compuesta por dos tapas de chapa metálica de 3 mm de espesor, cuadradas de 200 mm de lado. La tapa inferior posee orejas en cada lado, por donde la base puede ser fijada a la superficie de trabajo. La tapa superior tiene un agujero de 5 mm
De diámetro por donde pasa el eje que une el eslabón número 1 con la base. Ambas tapas de la base están unidas entre sí por ocho tubos de aluminio de 6,35 mm (1/4 pulgada) de diámetro y 74 mm de altura, soldados a las tapas; dejando la base con una altura total de 80 mm. Ver Figura 3.14.

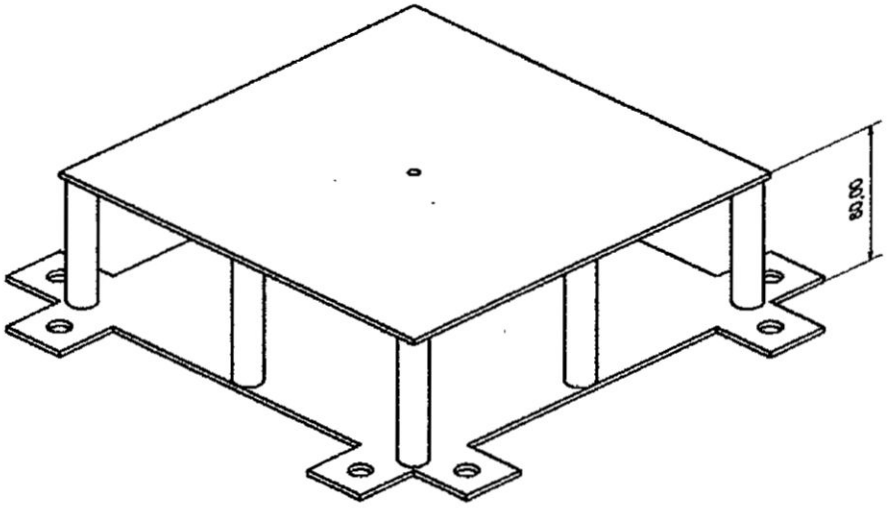


Figura 3.15 Base del robot de 3 G.D.L.

Al eslabón número uno se suelda un eje de aluminio de 5 mm de diámetro y 85 mm de longitud, por medio del cual será unido a la base del

manipulador. Una vez soldado el eje al eslabón número uno, se introduce por el agujero de la tapa superior del eslabón. El eje será fijado a la base por medio de dos rodamientos fijos a la tapa superior e inferior, que permitirán el movimiento rotacional de los eslabones minimizando el roce. Estos rodamientos serán colocados cada uno en una carcasa, que serán las que se fijen mediante soldadura a la tapa superior e inferior. Ver Figura 3.15.

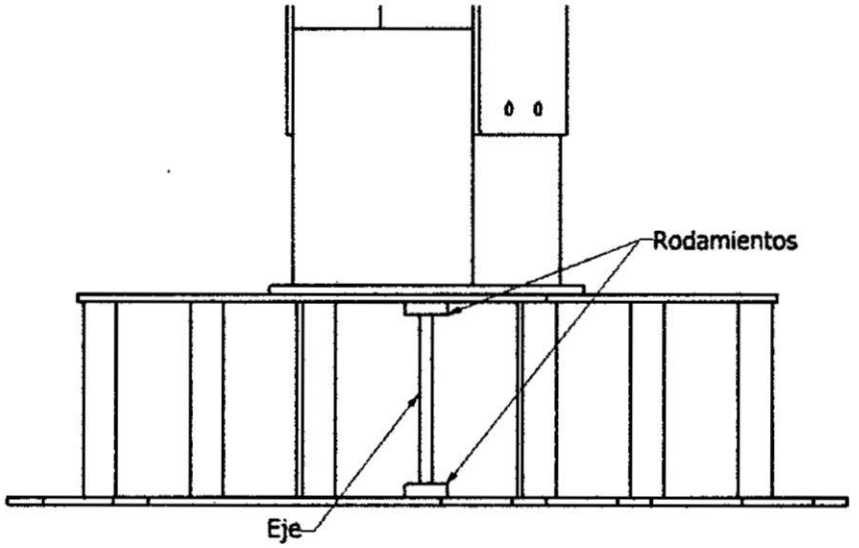


Figura 3.16 unión de la base con el eslabón 1

A medida que se diseñan las piezas del manipulador, se van digitalizando tridimensionalmente en un programa CAD, de manera de hacer análisis y comprobaciones del mecanismo mientras se diseña, y así minimizar errores durante la fase de diseño. Una vez digitalizadas cada una de las piezas de los eslabones, se realiza el ensamblaje de las mismas, y se comprueba el rango de movimiento de las mismas. Ver Figura 3.16, Figura 3.17 y Figura 3.18.

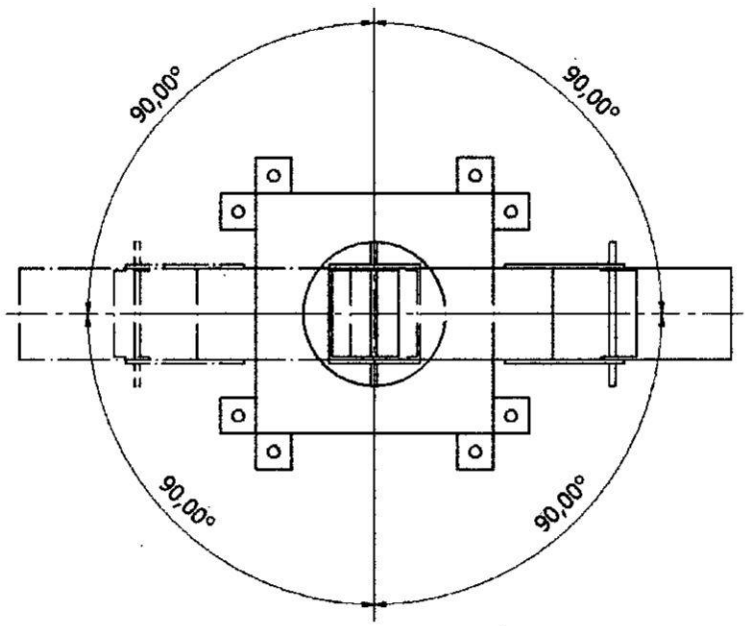


Figura 3.17 Movimiento del eslabón 1

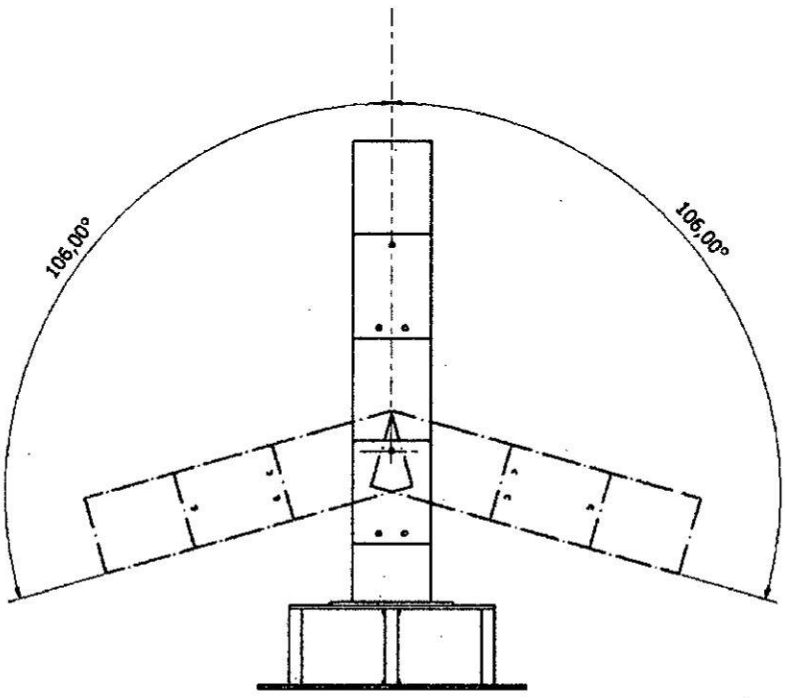


Figura 3.18. Movimiento del eslabón 2.

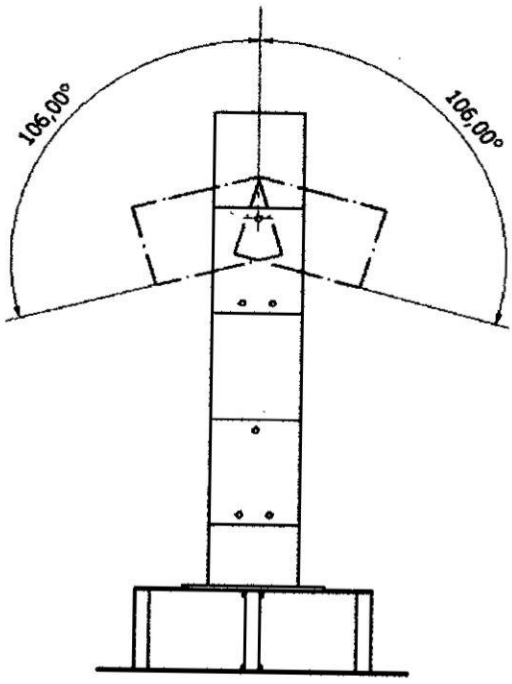


Figura 3.19. Movimiento del eslabón 3

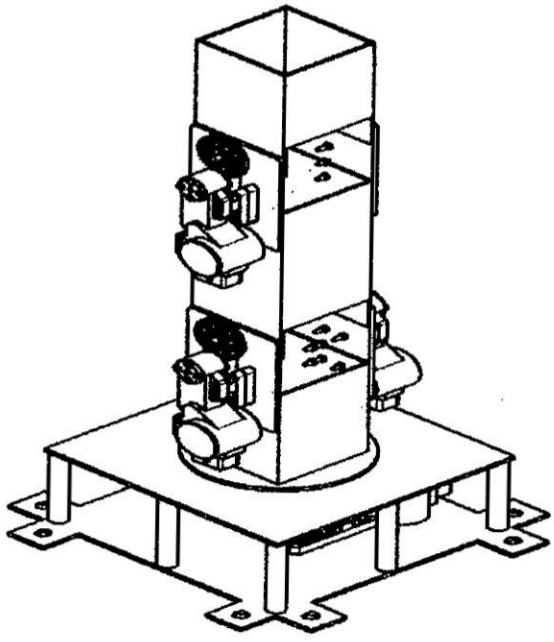


Figura 3.20 Robot de 3 G.D.L ensamblado digitalmente.

En el análisis de fuerzas que hay en cada articulación el autor observó que el hombro del robot debe ejercer más par que las otras articulaciones. Para el cálculo del par el autor tomó la decisión de realizar un brazo robótico que tuviera una extensión máxima de 33[cm], apartir de la segunda articulación hasta terminar el efector final o pinza como se ve en la Figura 3.12.

Para realizar el cálculo del par se debe saber lo siguiente. Cuál es el par máximo que puede soportar el servomotor. El par se calcula como M=F*d. donde,

M: es el momento de torsión o par.

F: La fuerza que debe soportar o en este caso es el peso del brazo.

d: la distancia de todo el brazo.

Realizando los cálculos necesarios del Robot, sabiendo que el servomotor puede soportar como máximo 1.5 [N*m]. El brazo construido en material de aluminio tiene una masa de 257[g] por lo que su peso es de 2.521[N], el brazo tiene una distancia total de 0.33 [m]. El par resultante es:

$$M = (2.521[N])(0.33[m]) = 0.8319[Nm] \qquad (3.1)$$

En la ecuación (3.1) se observa el par máximo que ejerce el brazo cuando se encuentra en posición horizontal. Esto equivale a casi la mitad del par que soporta el motor. Esto permite dar a conocer que la pinza del robot puede sostener un objeto con masa máxima de 200[g]. Si el servomotor se le aumenta una carga de 200[g] ejercería un par aproximado a 1.5 [N*m].

3.4 DISEÑO DEL SOFTWARE

Durante el proceso de este trabajo se optó por usar MATLAB & SIMULINK, ya que se pueden crear aplicaciones mediante el uso de lenguaje escrito y gráfico. La comunicación entre la computadora y los servomotores se utilizó el microcontrolador Atmega 2560, ya que el uso de la USB2Dynamixel para MATLAB está restringido por la librería y sólo permite mandar y obtener una respuesta a la vez.

Para poder programar el microcontrolador Atmega 2560 se utilizó el software Arduino, este software se basa en el lenguaje de programación C. Actualmente existe una librería en Arduino desarrollada para poder controlar los servomotores AX-12 y AX-18 de Dynamixel, la desventaja fue que no contenia todas las funciones necesarias para el proyecto. Por lo tanto, a la librería se le agregaron funciones para poder mover todos los motores al mismo tiempo.

El enlace entre los motores y el microcontrolador se realiza mediante la comunicación UART, como se muestra en la Figura 3.8.

IV. VARIABLES E HIPOTESIS

4.1. DEFINICIÓN DE LAS VARIABLES

Para los fines de este trabajo consideraremos las siguientes variables relacionadas entre sí para el establecimiento de un seguimiento a trayectorias, las cuales serán más precisas en la medida que se realice un correcto ajuste al controlador PID en el entorno SIMULINK MATLAB.

a. Variables Independientes.

- ✓ Modelamiento y Análisis de un robot de 3 G.D.L

b. Variables Dependientes.

- ✓ Comprensión de los Algoritmos de control

4.2. HIPÓTESIS

4.2.1 Hipótesis general:

Mediante la implementación de la cinemática directa y cinemática inversa mediante el uso de interfaces en MATLAB, se podrá modelar y analizar un robot de 3.G.D.L para la comprensión de algoritmos de control en estudiantes de Ingeniería Electrónica.

4.2.2 Hipótesis especifica:

Mediante el Modelamiento de la cinemática directa e inversa usando MATLAB se permitirá modelar el controlador PID para el seguimiento de trayectorias utilizando un método de sintonización.

V. METODOLOGIA

5.1. TIPO DE INVESTIGACIÓN

Documental / Experimental vía simulación y "real".

De una manera general, las actividades a realizarse serán: Identificación del problema, Verificación de la documentación existente, selección del software adecuado para las simulaciones y finalmente una comprobación en un prototipo de Robot real para establecer un control a la trayectoria de sus ejes.

Los datos resultantes serán obtenidos por simulación con el software MATLAB, este software será el que permita en su entorno SIMULINK sintonizar el controlador creado ahí, permitiendo así controlar la trayectoria del robot de 3 GDL.

5.2. DISEÑO DE LA INVESTIGACIÓN

La estrategia a ser seguida será la siguiente:

1) Hacer un modelamiento para el robot de 3 GDL en el entorno SIMULINK MATLAB estableciendo su modelo cinemático directo e inverso así como su modelo Dinamico antes de ponerlo a prueba con el robot real.
2) mediante las interfaces de comunicación y el robot adquirido se establecerá un controlador en el entorno SIMULINK MATLAB para establecer un control de las articulaciones del robot y un seguimiento de su trayectoria.

5.3. POBLACION Y MUESTRA

No aplicable.

5.4. TECNICAS E INSTRUMENTOS DE RECOLECCION DE DATOS

Como mencionado en el ítem anterior, la principal herramienta utilizada será el software de MATLAB, la cual mediante el entorno de SIMULINK permitirá el desarrollo de su cinemática directa e inversa además de poder representar y configurar en el programa el modelo dinámico que de acuerdo a los datos del robot real, nos brindara una simulación de sus trayectorias, permitiendo también con ello reunir información del movimiento de cada una de sus articulaciones.

5.5. PLAN DE ANALISIS ESTADISTICO DE DATOS.

Los resultados obtenidos en la simulación y pruebas "reales" serán agrupados y analizados, para obtener una mejor sintonización de las variables del controlador PID, creado en el entorno de SIMULINK MATLAB.

VI ANALISIS DE RESULTADOS

Como se observó en el Capítulo 2 se realizó el análisis teórico del robot por el cual se desarrollaron simulaciones para verificar si el robot diseñado con las características deseadas se comporta de forma correcta en el espacio de trabajo, para evitar colisiones y daños en el equipo, robot real. Por lo que se decidió generar distintas pruebas para verificar que el robot se desplace de la manera correcta.

Para la implementación del robot se generaron 3 pruebas:

- Implementación del tiempo de muestreo en el microcontrolador.
- Cinemática directa en el robot.
- Cinemática inversa en el robot.
- Implementación del Control PID.
- Implementación del seguimiento de trayectoria circular.

- Implementación del seguimiento de trayectoria mediante parametrización.

6.1 IMPLEMENTACIÓN DEL TIEMPO DE MUESTREO

El tiempo de muestreo, es el tiempo que transcurres entre dos mediciones consecutivas, es fundamental para la adquisición de datos y se suele expresar en frecuencia. Siempre que se desea medir un sistema, se necesita que nuestra frecuencia de muestro sea superior a la frecuencia del sistema.

El tiempo de muestreo se calculó de forma experimental utilizando SIMULINK y el programa del microcontrolador. En el microcontrolador se realizó la siguiente operación:

$$Tiempodemuestreo = Tiempo_{actual} - Tiempo_{anterior} \quad (6.1)$$

Esta prueba soló se realizó para obtener una aproximación del tiempo de muestreo. En el programa del microcontrolador se definen las variables del Tiempo, el tiempo actual se adquiere al final del programa y el tiempo anterior al inicio del mismo, el tiempo de muestreo se calcula después de haber obtenido el tiempo actual. El microcontrolador recibe la información necesaria enviada desde una computadora y al final regresa los datos obtenidos de los sensores junto con el tiempo de muestreo.

El valor experimental obtenido de la ecuación (6.1) es:

T iempodemuestreo = 0.7 [ms]

6.2 CINEMÁTICA DIRECTA EN EL ROBOT

Para comprobar el análisis realizado respecto a la cinemática directa del capítulo 2, se desarrolló una interfaz en GUI MATLAB y el microcontrolador para poder desplazar el robot.

Para su comprobación se le asignaron a $\theta_1 = 0, \theta_2 = \frac{\pi}{3}$ y $\theta_3 = -\frac{\pi}{2}$, dando como resultante $X_d = 20cm$ $Y_d = 0cm$ $Z_d = 12.6\,cm$ en la interfaz

desarrollada se asignaron los valores de θ_1 para la cintura, θ_2 para el hombro y θ_3 para el codo, como se muestra en la figura 6.1

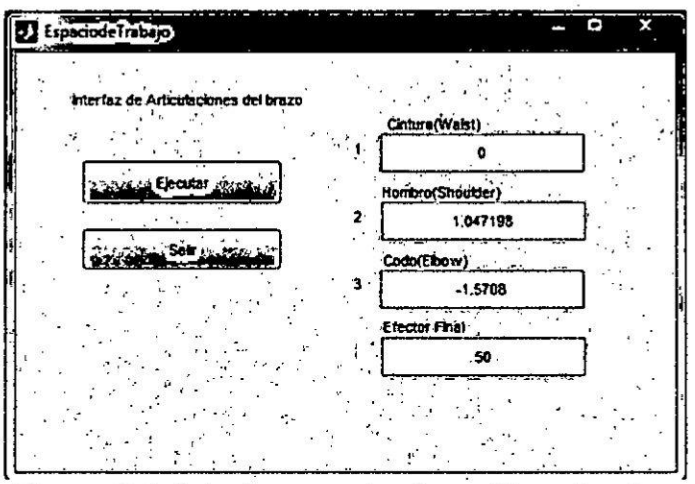


Figura 6.1: Interfaz para la cinemática directa

El resultado de la interfaz se implementa en el robot real, por lo que de la prueba se obtiene lo mostrado en la las medidas correspondientes a X_d, Y_d y Z_d del robot y se verificó que las distancias obtenidas del efector final con respecto al sistema base eran las mismas, que se asignaron.

6.3 CINEMÁTICA INVERSA EN EL ROBOT

Para comprobar el análisis realizado respecto a la cinemática inversa del capítulo 2, se desarrolló una interfaz en GUI MATLAB, donde a diferencia de la cinemática directa a este se le asignan las posiciones deseadas del efector final. Para su comprobación se le asignaron a $X_d = 12cm$ $Y_d = 12cm$ $Z_d = 6.7\,cm$, dando como resultante $\theta_1 = \frac{\pi}{4}$, $\theta_2 = \frac{5}{18}\pi$ y $\theta_3 = -\frac{17}{30}\pi$ en la interfaz desarrollada se asignaron los valores de X_d, Y_d y Z_d como se muestra en la Figura 6.2.

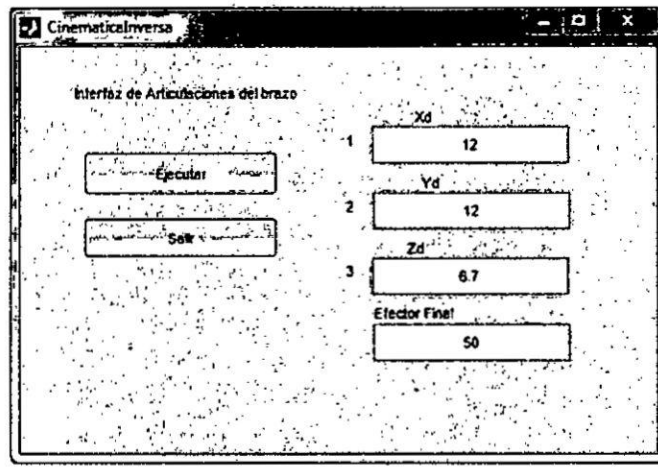


Figura 6.2: Desplazamiento del robot con cinemática inversa

El resultado de la *interfaz* se implementa en el robot real, por lo que de la prueba se obtiene las medidas correspondientes a θ_1, θ_2 y θ_3 del robot y se verificaron los ángulos correspondientes de cada articulación.

6.4 IMPLEMENTACIÓN DEL CONTROL PID

Como se observa en el Capítulo 2, en la simulación se utilizó un controlador PID con compensación de gravedad en el sistema en lazo cerrado del Robot para observar su comportamiento.

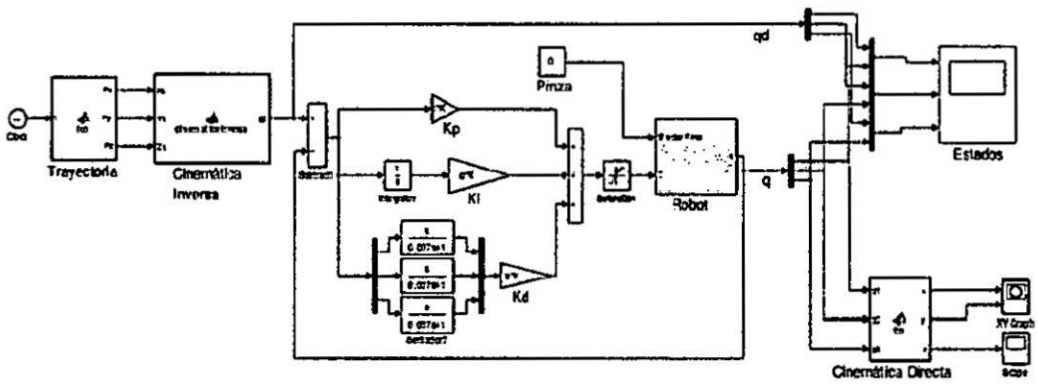


Figura 6.3: Diagrama de bloques para el control PID del Robot

En la Figura 6.3 se observa el diagrama de bloques para el control PID del Robot, este diagrama de bloque es similar al de la simulación, la diferencia es que en este no se toma el modelo dinámico del Robot a menos que se deseé implementar otro tipo de controlador.

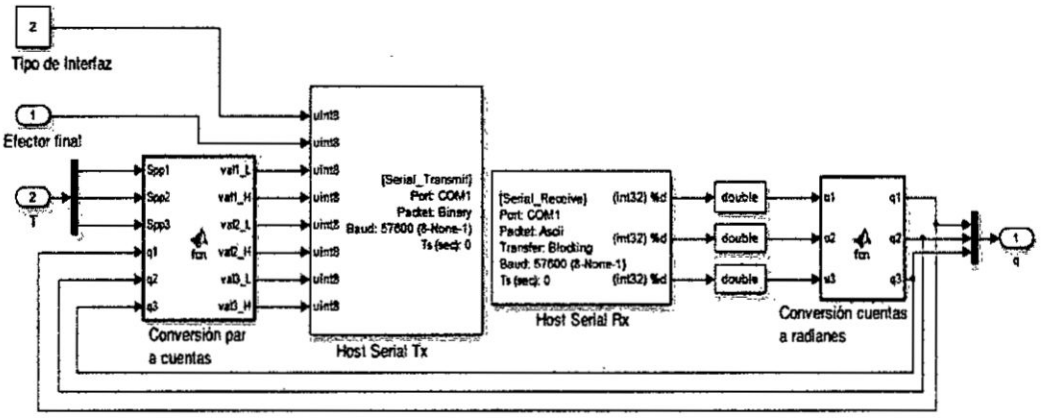


Figura 6.4: Bloque de comunicación del Robot

En la Figura 6.3 hay un bloque cuyo nombre es Robot, lo que tiene internamente este bloque se puede apreciar en la Figura 6.4, este bloque está compuesto de lo siguiente:

- Conversión de par a cuentas, se encarga de adquirir los datos en Nm y los convierte a la resolución del servomotor. Se sabe que los motores tienen un par máximo de 1.5 [Nm], por lo que para ese par corresponde la cuenta 1023 en sentido antihorario, y para el sentido horario corresponde 2047. Para cuando el par es 0 [Nm], corresponde la cuenta 0 en sentido antihorario, y para el sentido horario corresponde 1024.
- Conversión de cuentas a radianes, la lectura de la posición articular va de 0[rad] a 5.236[rad](en grados es de 0° a 300°),para cuando está ubicado en 0[rad] la cuenta es cero y para cuando se ubica en 5.236[rad] la cuenta es 1023 sin importar el sentido de giro.
- bloques de conversión de entero a double, los datos recibidos del microcontrolador se toman como entero, por lo que se adquiriran

valores de 0 a 1023 enteros, se convierte a double porque se desea el valor en radianes.

- Bloque de Host Serial Tx, este bloque sirve para enviar todos los datos necesarios al microcontrolador.
- Bloque de Host Serial Rx este bloque sirve para recibir todos los datos de las posiciones articulares del microcontrolador
- Tipo de interfaz, este bloque se habilita en 2 para el caso de Simulink, debido a que la comunicación se realiza en lazo cerrado.
- Efector Final, se habilita con 0 para cuando está abierta la pinza y 100 para cerrar completamente la pinza, este dato es en enteros.
- T, esta es la entrada del par de cada motor.

El controlador PID obtenido en la simulación se implementó en el Robot construido, pero el resultado no fue igual que el simulado. Para obtener los valores de las ganancias experimentalmente para el controlador PID, existen distintos métodos de sintonización. En el caso de este trabajo se utilizó el método de sintonización de Ziegler - Nichols de oscilaciones continuas.

6.4.1 Método de sintonización de Ziegler - Nichols de oscilaciones continúas

El método tiene la ventaja de que no es necesario conocer la planta del sistema para poder sintonizarlo. Su sintonización se realiza en lazo cerrado, este procedimiento es válido solo para plantas estables a lazo abierto. La sintonización de este controlador PID para cada articulación se hace independiente y es por regulación, es decir, en la sintonización a la articulación que se analizará se le asigna una posición articular deseada de 1 [rad].

Para la sintonización del control PID, en el caso de este Modelamiento primero se sintoniza la tercera articulación, ya que esta articulación no depende del funcionamiento de la segunda y la primera articulación. Después se sintoniza la segunda articulación, para este caso se habilita

el control PID de la tercera articulación dándole la posición deseada con la que se sintonizó. Finalmente se sintoniza la primera articulación en este caso se habilita el control PID de la segunda y tercera articulación con la posición articular deseada que se sintonizó, ya que si no se le ejerce el control a las otras articulaciones no es posible sintonizar esta articulación, porque el efector final puede tener una fuerza de fricción con la base del espacio de trabajo.

Para la sintonización de cada articulación se realizaron los siguientes pasos:

1. Sólo se utilizó la acción proporcional, comenzando desde un valor de ganancia pequeño, se incrementó la ganancia proporcional hasta que la salida del sistema presentara oscilaciones. La ganancia obtenida es conocida como *Kc*, representa la ganancia crítica a la que puede llegar el sistema sin el comportamiento de la acción integral y derivativa.
2. Las oscilaciones tienen un período continuo que se le conoce como P_{cr}, como se observa en la Figura 6.5.

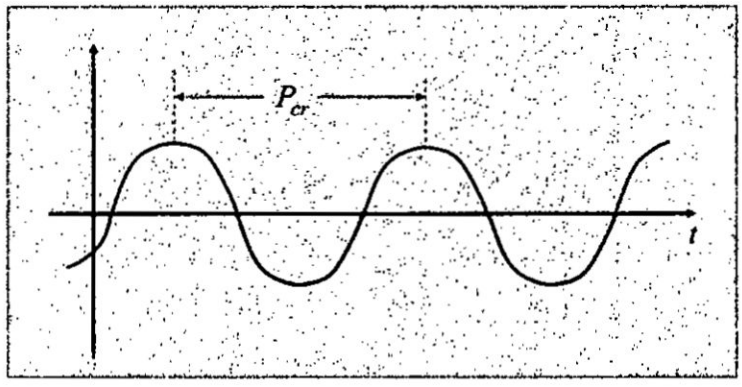


Figura 6.5: Oscilaciones continúas

3. Se calcularon las ganancias del controlador, respecto a la Tabla 6.1.

Controlador	K_p	τ_i	τ_d
PID	$0.6K_c$	$0.5P_{cr}$	$0.125P_{cr}$

Tabla 6.1 Sintonización de PID

En la tabla 6.1 se obtiene la ganancia proporcional, para obtener la ganancia integral y derivativa se calculan de la siguiente forma:

$$K_i = \frac{K_p}{\tau_1} \qquad (6.2)$$

$$K_d = K_p\tau_d \qquad (6.3)$$

Las ganancias K_i y K_d dependen de la ganancia proporcional como se observa en la ecuación (6.2) y (6.3). En este trabajo se obtuvieron los siguientes resultados tomando en cuenta las ecuaciones (6.2) y (6.3) y las ecuaciones de la Tabla 6.1.

Para la primera articulación se obtuvo $K_c = 3.4$ y $P_{cr} = 13.2$ en base a estos valores se calcularon las ganancias del control PID que dieron como resultado: $K_p = 2$, $K_i = 0.3$ y $K_d = 1.65$.

Para la segunda articulación se obtuvo K_c= 8.81 y P_{cr}= 5.55, en base a estos valores se calcularon las ganancias del control PID que dieron como resultado: K_p = 5.286, K_i= 1.9 y K_d = 3.66.

Para la tercera articulación se obtuvo K_c = 8.3 y P_{cr}= 7.8, en base a estos valores se calcularon las ganancias del control PID que dieron como resultado: $K_p = 4.98, K_i = 1.27$ y $K_d = 4.85$.

En base a las ganancias del controlador PID calculadas para cada articulación se probaron en el Robot y se observó su comportamiento. El comportamiento que presento cada articulación del Robot se observaron pequeñas oscilaciones, por lo que se ajustaron las ganancias K_i y K_d, tal que las oscilaciones disminuyeran lo más posible o desaparecieran.

6.4.2 Método de diferenciación

En la implementación del microcontrolador con MATLAB, es necesario un método de derivación, debido a que no se tiene sensor de velocidad. Los servomotores AX-12 pueden entregar datos de velocidad angular, pero tienen la desventaja que usan un método muy sencillo conocido como diferenciación numérica. La diferenciación numérica se calcula como:

$$m_s = \frac{f(t+h)-f(t)}{(t+h)-t} \qquad (6.4)$$

En la ecuación (6.4) representa un cambio en la posición con respecto al tiempo y h representa un incremento. Para el caso del proyecto, el resultado obtenido es la velocidad promedio de la articulación analizada. Sin embargo, se observó que el cálculo de la velocidad por diferenciación numérica no presenta un buen desempeño. Para la resolución del potenciómetro del motor, la velocidad calculada por este método es imprecisa para velocidades bajas y altas.

Se optó entonces por estimar la velocidad angular por medio de un filtro estable paso altas de primer orden con grado relativo cero. Este método de diferenciación aproximada, tiende un mejor desempeño que con el enfoque tradicional de diferenciación numérica, por lo que es comúnmente utilizada en aplicaciones que requieren regulación de velocidad. Se puede representar como la ecuación (6.5).

$$G(s) = \frac{a*s}{b*s+1} \qquad (6.5)$$

En la ecuación (6.5) se puede observar la interpretación de la derivada sucia. Para a = 1, y b= tiempo de muestreo. En el proyecto se utilizó la derivada sucia para poder realizar la acción derivativa del controlador PID. El resultado de la derivada aproximada en el robot se observa en la ecuación (6.6).

$$G(s) = \frac{s}{0.007\,s+1} \qquad (6.6)$$

6.5 IMPLEMENTACIÓN DEL SEGUIMIENTO DE TRAYECTORIA CIRCULAR

En el Capítulo 2 se puede observar en la generación de trayectorias, el análisis teórico del seguimiento de trayectoria circular. El diagrama de bloques del Robot real en SIMULINK cambia con respecto al de la simulación, debido a que se incluye el tiempo de muestreo y el método de derivación, este se puede observar en la Figura 6.5.

Para el seguimiento de trayectoria, se tomaron en cuenta los valores calculados en el método de sintonización del controlador PID. Se ejecutó el programa para que el robot real realizara la trayectoria y en SIMULINK se adquirieron todos los datos mientras se estuviera ejecutando. De los datos obtenidos se graficaron los estados que representan cada una de las posiciones articulares con respecto al tiempo, como se observa en la Figura 6.8.

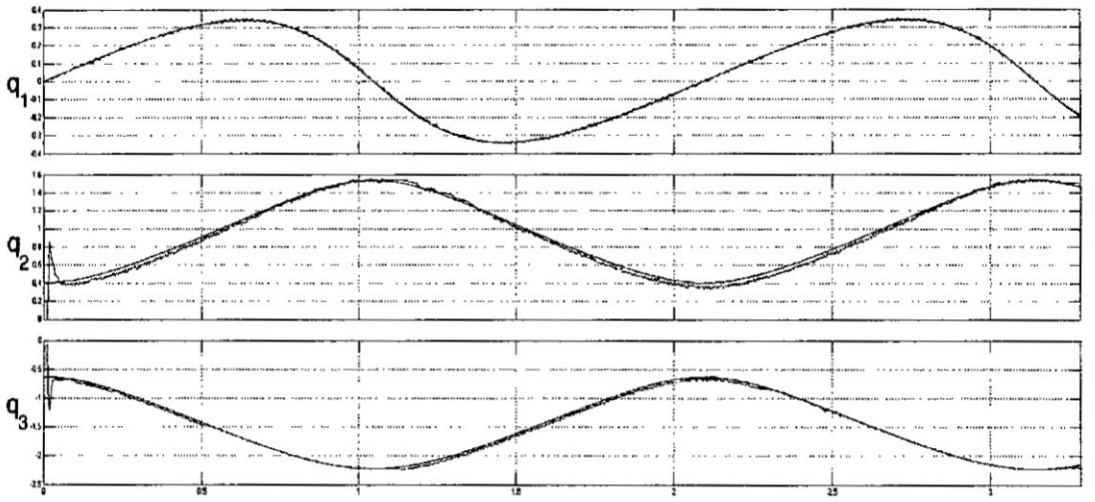


Figura 6.6: Comportamiento de los estados del Robot de 3GDL

En las Figuras 6.6 se puede observar el comportamiento de cada articulación, la articulación q_1 sigue la trayectoria relativamente bien ya que en esta articulación no hay efectos de la gravedad. Para el caso de las articulaciones q_2 y q_3 , en estas si hay efecto de la gravedad por lo que se le dio una compensación de gravedad al robot para poder obtener un mejor comportamiento del mismo, esta compensación es la que se muestra en la ecuación (2.30).

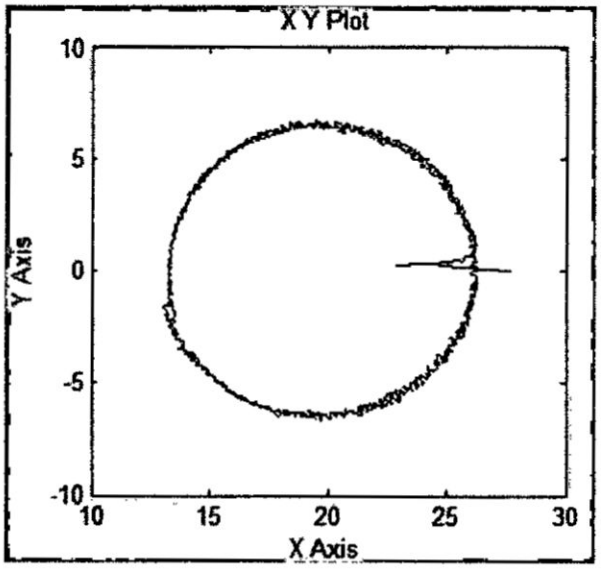


Figura 6.7: Grafica de la trayectoria en el Plano XY

En la Figura 6.9 se observa todo el desplazamiento generado en el Plano XY, por lo que se observan pequeños picos, y esto se debe a las ganancias proporcionadas en el robot y el efecto de la gravedad en las articulaciones q_2 y q_3.

6.6 IMPLEMENTACIÓN DEL SEGUIMIENTO DE TRAYECTORIA MEDIANTE PARAMETRIZACIÓN

El análisis teórico para la generación de una trayectoria mediante parametrización se puede observar en el Capítulo 2 en la generación de trayectorias. A diferencia del diagrama de bloques del seguimiento de trayectoria circular este cambia un poco, debido a que se desea que el brazo robótico sujete un objeto y lo desplace al lado contrario.

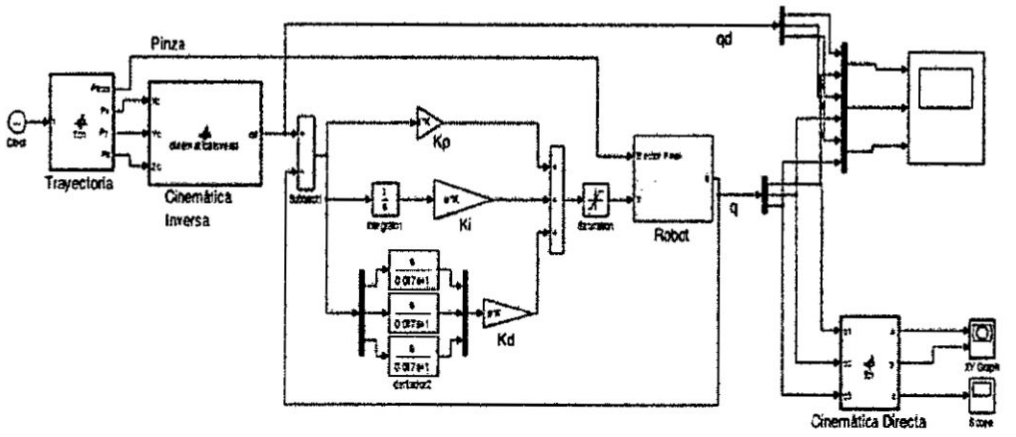


Figura 6.8: Diagrama de bloques para el control PID del Robot con movilidad en el efector final

En la Figura 6.8 se observa que el bloque de trayectoria tiene una entrada más correspondiente al efector final. Para este caso se le da una condición inicial al efector final para poder sujetar la pieza y una condición final para soltar la pieza como se comentó en el Capítulo 2. Su implementación fue similar a la del seguimiento de trayectoria circular, el

problema fue que presento un poco más de oscilaciones en sus posiciones articulares.

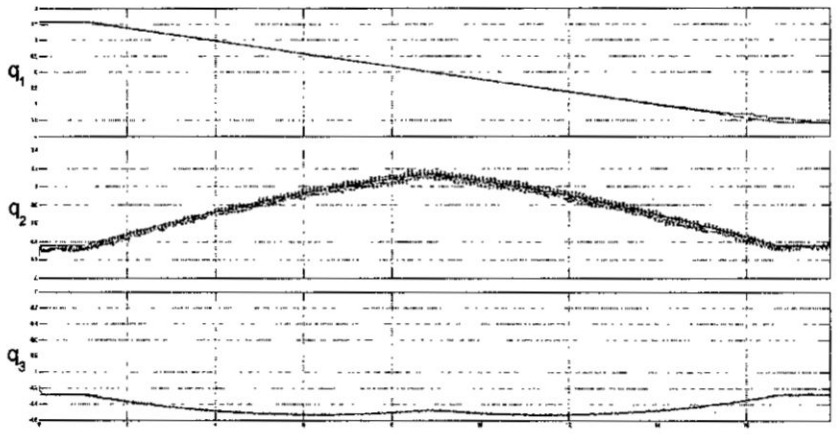


Figura 6.9: Comportamiento de los estados del Robot real con movilidad en el efector final

Como se puede observar en la Figura 6.9 las posiciones articulares de $q_1 y\, q_3$ siguen adecuadamente la trayectoria del Robot, en q_2 se tiene que ajustar mas las ganancias del controlador para que tenga un mejor comportamiento.

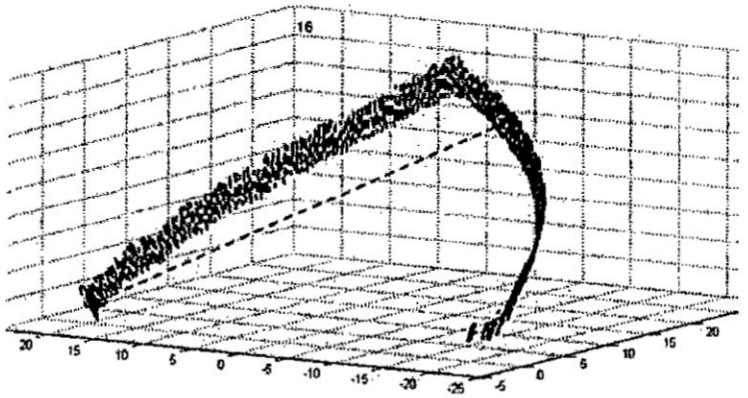


Figura 6.10: Grafica del seguimiento de trayectoria mediante parametrización del Robot real con movilidad en el efector final.

En la Figura 6.10 se observa la trayectoria que generó el Robot en tiempo real, el problema que se obtuvo aquí fue las posiciones articulares que tomo la segunda articulación por lo que generó muchas oscilaciones en su movilidad.

VII CONCLUSIONES

Finalmente luego de todo el trabajo realizado se comprueban los objetivos marcados en un inicio:

- ✓ Para el control del robot para seguimiento de trayectorias es necesario como paso previo conocer el modelo cinemática y dinámico del robot, así como de sus parámetros físicos para ello se usó el entorno de MATLAB.
- ✓ A través del presente trabajo se puede comprobar que con un control PID basado en Torque computado, es decir considerando las fuerzas que interactúan con el robot, como lo son las inercias, fuerzas de Coriolis, centrífugas y el efecto de la gravedad sobre cada uno de los eslabones, la trayectoria ejecutada por el robot se asemeja mucho a la trayectoria deseada.
- ✓ Sin este controlador PID el motor no da el torque que necesita cada una de las articulaciones del robot y por lo tanto no sigue la trayectoria deseada.
- ✓ Para objetivos didácticos se adquirió un modelo de robot de 3 G.D.L real, cuyos parámetros de diseño mecánico fueron vasados en las pautas dejadas por su diseñador. Este modelo de robot sirvió para comprobar mediante una interfaz desarrollada en MATLAB la importancia de la sintonización de las variables del controlador PID y por ende facilitar con una demostración la importancia de los algoritmos de control.
- ✓ Finalmente la experimentación se logró mediante el desarrollo de interfaces graficas que permiten obtener las posiciones articulares y la posición del efector final realizando una implementación en lazo abierto en el Robot y el desarrollo de trayectorias mediante la utilización de un controlador PID en lazo cerrado. En este caso se observaron algunos problemas, cuando se analiza su comportamiento en el Robot, el Robot presenta un pequeño decaimiento.

VIII RECOMENDACIONES

1. La cinemática implementada en esta Tesis fue la de generación de trayectorias punto a punto con un movimiento simultáneo de ejes. Por ello Definir trayectorias deseadas por medio de asignación de puntos se debe hacer con mucha destreza y exactitud para el mejor desempeño del brazo robótico.
2. El modelo dinámico es necesario para poder simular el comportamiento del robot, y aproximarse al control requerido para manejar las trayectorias en el robot real.
3. Para implementar estrategias de control es muy importante tener en cuenta todas las fuerzas que interactúan con el robot, como lo son las inercias, fuerzas de Coriolis, centrífugas y el efecto de la gravedad sobre cada uno de los eslabones. Por ello para tener todas estas consideraciones en el modelo Dinámico del Robot de 3 G.D.L, se emplea el método de Euler-Lagrange.

IX REFERENCIAS BIBLIOGRAFICAS.

- Andueza, L. (2008). Diseño de un manipulador robótico con tres grados de libertad con fines educativos. Mérida: Universidad de los Andes.
- Diccionario de la Real Academia Española. Definición de robot [en línea]: http://dle.rae.es/?id=WYRlhzm
- Antonio Barrientos, Luis Felipe Peñín y otros, 2007. Fundamentos de Robótica. Cinemática del robot.
- Departamento de Arquitectura y Tecnología de Computadores, Universidad de Sevilla. Tema 4, parte 2. Coordenadas homogéneas y matriz de transformación homogénea [en línea]: http://icaro.eii.us.es/descargas/tema_4_parte_2.pdf
- [Siciliano, B.(2010). Robotics Modelling, Planning and Control. En L. Sciavicco, L. Villani, G. Oriolo (Eds.). Londres: Springer- Verlag
- Lopez R., Mauro G.(2014). Puesta en marcha del robot Omni Phantom de Sensable. México: Universidad Nacional Autónoma de México
- Jazar, Reza N.(2007). Theory of Applied Robotics Kinematics, Dynamics and Control. Second Edition, Denavit-Hartenberg Notation(pp. 233-242). Londres: Springer- Verlag
- Francisco (2013). Grados de Libertad de un Robot. Fecha de consulta: 12 de Enero del 2017. http://coparoman.blogspot.mx/2013/05/12-grados-de-libertad-de-un-robot.html
- González, V. (2008). Estructura de los Robots Industriales. Fecha de consulta: 18 de enero del 2017. http://platea.pntic.mec.es/vgonzale/cyr_0708/archivos/_15/ Te-ma_5.4.htm
- ROBOTIS (2010). AX-12/ AX-12+ /AX-12A. Fecha de consulta: 18 de enero de 2017. http://support.robotis.com/en/product/dynamixel/ax_series/ dxl_ax_actuator.htm
- Castillo, Luis (2011). Memoria EEPROM. Fecha de consulta: 14 de enero del 2017. http://es.slideshare.net/be77ocas/eeprom-10170780

- Memoria RAM Fecha de consulta: 15 de enero del 2017. http://www.significados.com/memoria-ram/
- Microcontrolador. Fecha de consulta: 16 de enero de 2017. http://microcontroladores- e.galeon.com
- RoboPlus. Fecha de consulta: 18 de enero del 2017. http://support.robotis.com/en/software/roboplus_main.htm

ANEXO 2

APENDICES

Apéndice A

Configuración de Servomotores Dynamixel en RoboPlus

El Software RoboPlus tiene la funcionalidad de manipular y configurar los motores Dynamixel mediante el uso de un dispositivo de Robotis.

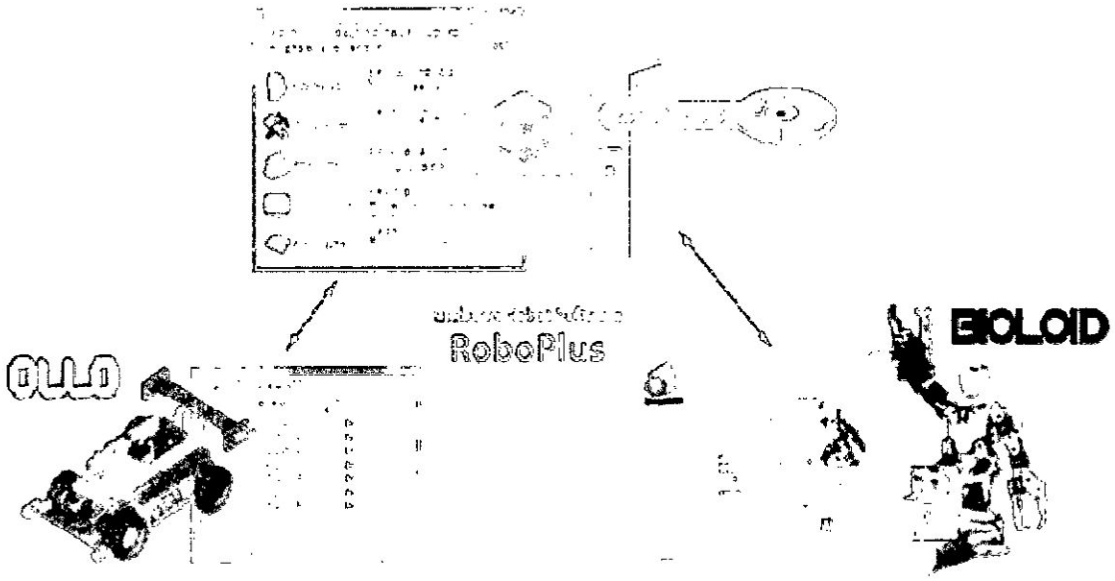


Figura A.1: RoboPlus

En este caso se utilizara una Usb2Dynamixel para la configuración de los motores del tema de Tesis: *MODELAMIENTO Y ANÁLISIS DE UN ROBOT DE 3 G.D.L PARA LA COMPRENSIÓN DE ALGORITMOS DE CONTROL, EN ESTUDIANTES DE INGENIERÍA ELECTRÓNICA.*

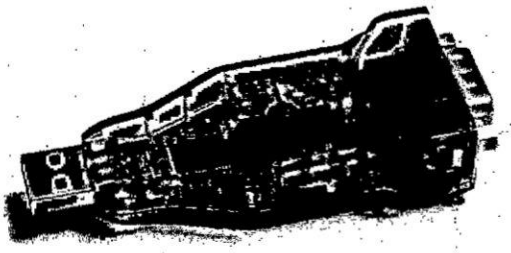


Figura A.2: USB2Dynamixel

Los pasos para realizar adecuadamente la configuración de los motores para no tener algún problema se explican a continuación:

Paso 1. Configurar el Usb2Dynamixel

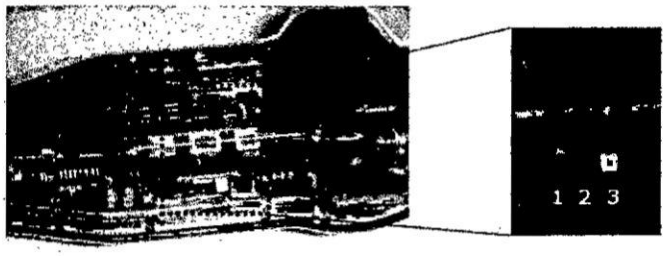


Figura A.3: Asignación de comunicación serial

En la Figura A.3 se observa la asignación del tipo de comunicación para el tipo de motor, en el caso de este trabajo de tesis el interruptor se debe seleccionar la comunicación TTL, en la Figura A.3 es la opción 1.

Paso 2. Verificar la conexión correcta de los motores.

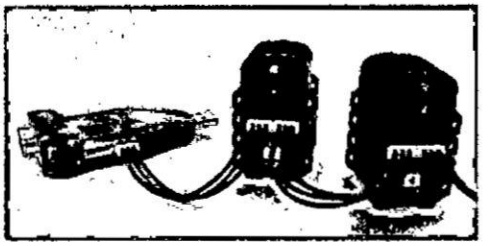


Figura A.4: Conexión de motores en Daisy Chain

Los motores Dynamixel AX tienen un conector con 3 puertos, este se conecta a la USB2Dynamixel, ***cuando se configura por primera vez los motores es recomendable que se configure primero un motor sin conectar los demás en serie, ya que si se configuran todos a la vez pueden ocurrir problemas De comunicación debido a que todos tienen la misma dirección ID, los motores están predeterminados con un ID:1.***

Para más información ver la página del Usb2Dynamixel: http://support.robotis.com/en/product/auxdevice/interface/usb2dxl_manual.htm.

Paso 3. Abrir la aplicación RoboPlus

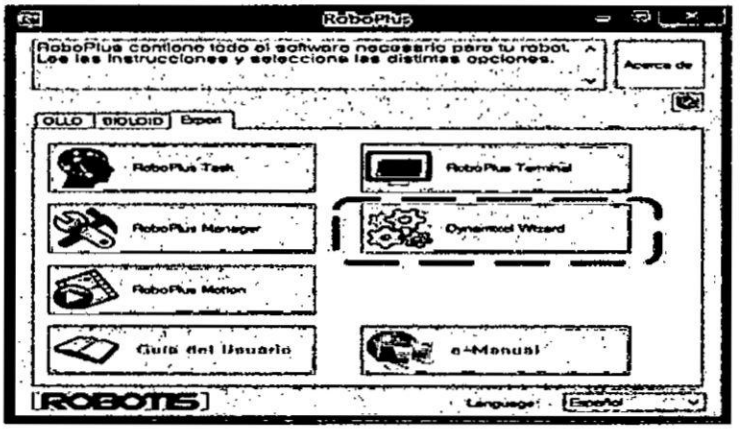
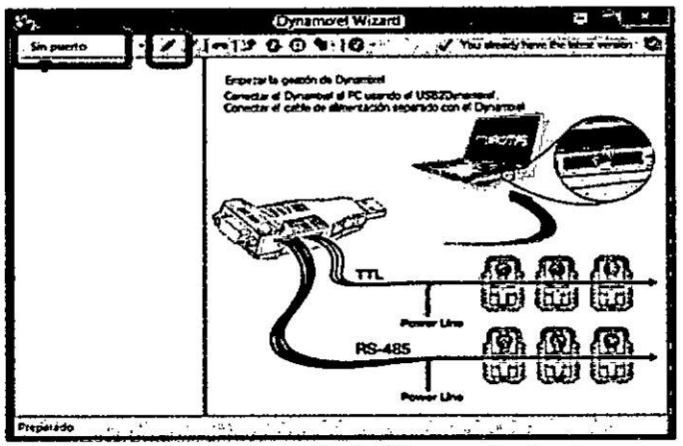


Figura A.5: Ventana Principal de RoboPlus

En la Ventana principal de RoboPlus se debe seleccionar la opción "Dynamixel Wizard" como se ve en la Figura A.5, ya que es donde se va a realizar la comunicación con los motores.

Paso 4. Conectarse a la Usb2Dynamixel

Figura A.6: Dynamixel Wizard



En este paso debe estar conectada la Usb2Dynamixel a la computadora y se debe saber cuál es el puerto COM al que está conectado el dispositivo.

En la ventana de la Figura A.6, ahí se va seleccionar el puerto en la opción que se enmarco de color rojo, se debe verificar que el puerto COM este asignado a la Usb2Dynamixel, de caso contrario no se enlazará a los motores.

Cuando se haya elegido el puerto COM correcto, seleccionar el botón que se en marco de color naranja en la Figura A.6.

Paso 5. Conectar motores para ser configurados

En este paso ya se debió de tener conectados los motores y deben de estar conectados a una fuente de alimentación de 12 V como máximo y mínimo 9 V.

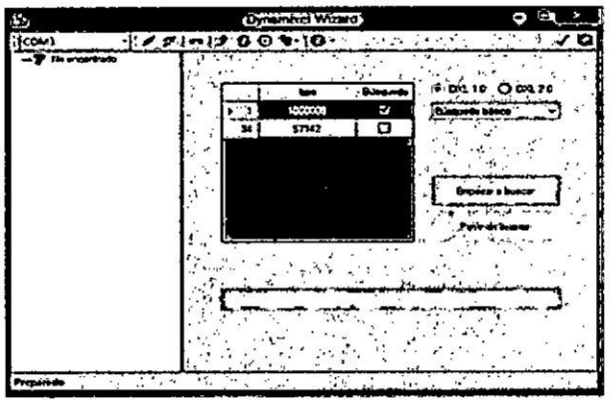


Figura A.7: Enlace de Configuración

En la Figura A.7 se selecciona la opción empezar a buscar, los bps asignados en este trabajo es el predeterminado que es 1000000 bps, por lo que al seleccionar el botón empezara a buscar los motores disponibles.

Paso 6. Selección de parámetros y motor a configurar

En este paso se explicará cómo asignar los IDs de los motores, para que no genere problemas de configuración cuando cada motor tenga asignado su propio ID como se explicó en el **Paso 2**.

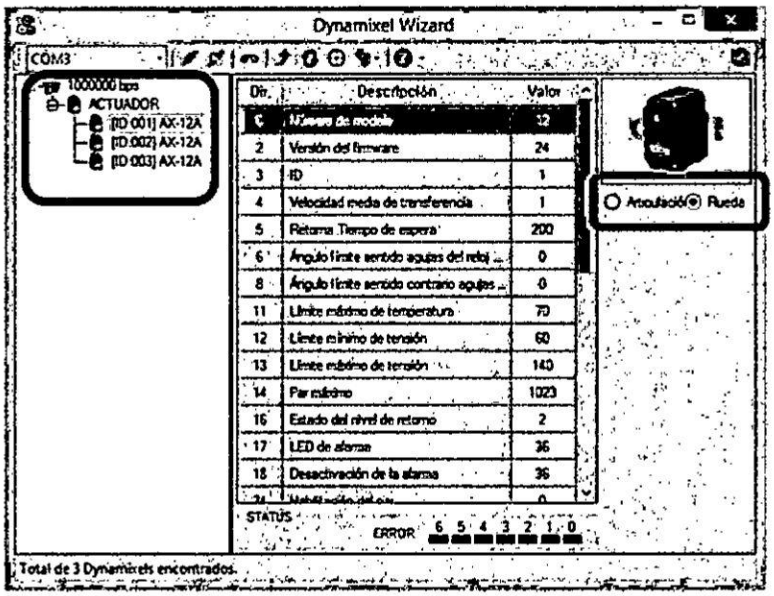


Figura A.8: Enlace de Configuración

En la Figura A.8 se enmarca de color rojo los motores con sus respectiva ID o nombre de identificación, esto permite que todos tengan nombres distintos para que no haya confusión en la comunicación. El ID se configura seleccionando la opción 3 o ID que se muestra en la Figura A.8, ahí va mostrar un botón y la opción de dirección para asignarle a los motores, si se configuran varios motores que tengan una identificación de números sucesivos empezando desde el No. 1 como se ve en el recuadro en rojo.

En la Figura A.8 se enmarca de naranja la modalidad de manejo del motor, ya que el modo articulación tiene la funcionalidad de manipular al motor como un servomotor, en el que se le asigna la posición y velocidad deseada para el motor.

Estos servomotores no se comunican por señal PWM, lo hacen mediante comunicación UART. El modo Rueda permite manipular al motor como un motor de DC.

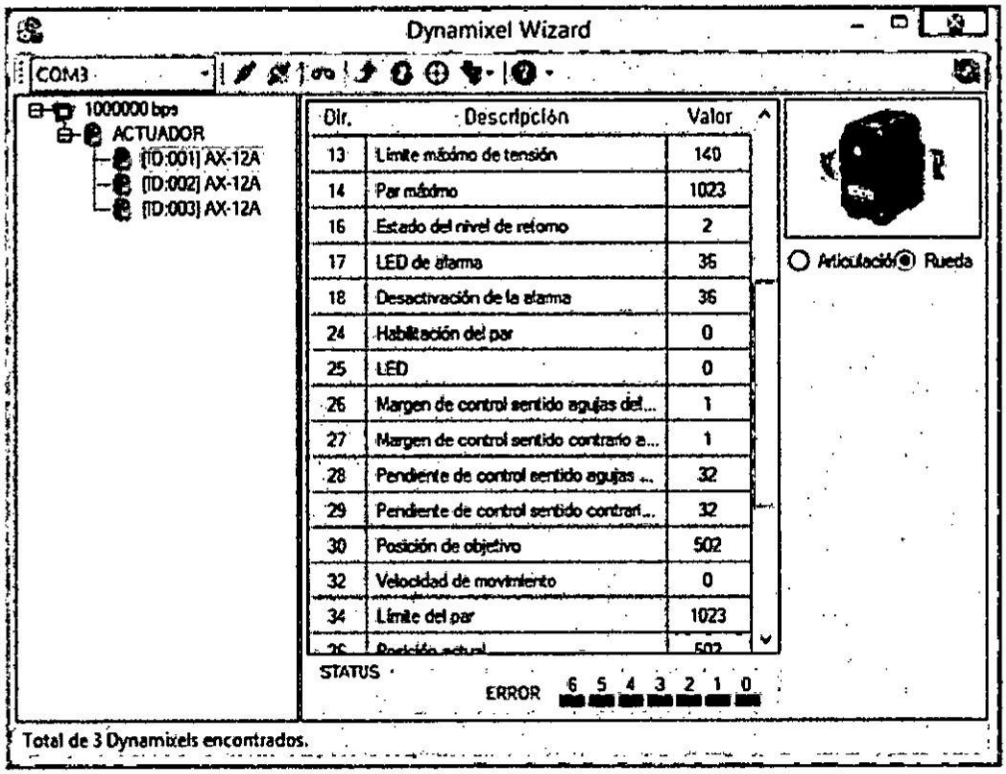


Figura A.9: Enlace de Configuración

En las Figuras A.8, A.9 y A.10 se muestran todos los parámetros que se deben de asignar para poder utilizar los motores. El único parámetro que cambia y no puede ser igual de cada motor es el 3 o ID. Para mayor información checar los manuales de Dynamixel. Las opciones 30 hasta la 46 no se toman en cuenta en

la configuración, ya que con estos puedes mover el motor o tomar lectura de los datos que te puede entregar cada uno de ellos.

Apéndice B

Comandos utilizados en la librería de Arduino DynamixelSerial1

La librería para Arduino se utilizaron los siguientes comandos.

1. **begin();**

   Descripción: Inicializa la comunicación serial en el Arduino.

   Sintaxis: begin(Baudrate, DataControl);. Esta predeterminado como begin(1000000,2);. Parámetros:

   Baudrate esta predeterminado en el Arduino como 1000000 bps

   DataControl esta predeterminado como 2, en donde este representa le puerto núm. 2 del Arduino.

   RxPin puerto de recepción de datos del Arduino.

   T xPin puerto de transmisión de datos del Arduino.

2. **setEndless().**

   Descripción: Habilita o Deshabilita el modo continuo en la rotación del servomotor, o sea, permite hacer girar el motor como un motor de DC donde solo necesitara el par de giro o como un servomotor en el que se le da la posición y velocidad a la que desea llegar.

   Sintaxis: setEndless(ID,Status);

   Parámetros:

   ID Numero de identificación para cada servomotor.

   Status el estado en el que se desea estar (ON o OFF). Para ON permite girar el eje del motor hasta que el usuario lo detenga. Para OFF permite girar el motor hasta la posición que se desea llegar con respecto al potenciómetro.

3. **torqueStatus().**

   Descripción: Habilita o deshabilita el torque en el servomotor.

Sintaxis: torqueStatus(ID,Status);

Parámetros:

ID Número de Identificación del Servomotor.

Status ON o OFF para activar o desactivar el torque.

4. **readPosition().**

Descripción: Lee la posición de uno de los actuadores que se deseen conocer. Sintaxis: readPosition(ID);

Parámetros:

ID Número de Identificación del Servomotor.

5. **SyncPos3.**

Descripción: Esta función permite desplazar 3 servomotores a posiciones deseadas. Este funciona para cuando el Endless esta desactivado.

Sintaxis:

syncPos3 (Position1_L, Position1_H, Position2_L, Position2_H, Position3_L, Position3_H);

Parámetros:

ID Para este caso deben de estar predeterminadas los ID's como 1, 2 y 3 para cada servo- motor.

PositionN_L (Position1_L, Position2_L y Position3_L) – Representa un valor de 0 – 255, este es una parte de la posición a la que se desea llegar.

PositionN_H(Position1_H, Position2_H y Position3_H) – Representa un valor de 0-3.

PosicionFinal El servomotor AX-12 puede llegar hasta una posición de 0300 grados o de 0 a 1023 cuentas. La posición es una suma entre el dato PositionN_L y PositionN_H.

PosicionFinal= 256*PosicionN_H + PosicionN_L

Por lo que si se desea llegar hasta 300 grados o 1023 cuentas.
PosicionFinal=256*3+255 = 1023
Por lo que si se desea llegar hasta 150 grados o 512 cuentas.
PosicionFinal=256*2+0 = 512

6. **SyncTorque3.**

Descripción: Esta función permite desplazar 3 servomotores al torque deseado. Este funciona para cuando el Endless está activado.

Sintaxis:

SyncTorque3(Speed1_L, Speed1_H, Speed2_L, Speed2_H, Speed3_L, Speed3_H);

Parámetros:

ID Para este caso deben de estar predeterminadas los ID's como 1, 2 y 3 para cada servo- motor.

SpeedN_L (Position1_L, Position2_L y Position3_L) Representa un valor de 0 255, este es una parte de la posición a la que se desea llegar.

SpeedN_H(Position1_H, Position2_H y Position3_H) Representa un valor de 0 7.

ParFinal El servomotor AX12 puede llegar hasta un par de 0 a 1.5 Nm o de 0 a 1023 cuentas para el sentido anti-horario. Y de 1024 a 2047 en el sentido horario, donde 1024 representa un par de 0 y 2047 representa un par de 1.5 Nm aproximadamente. El Par Final es una suma entre el dato SpeedN_L y SpeedN_H.

ParFinal=256*SpeedN_H+SpeedN_L.

Por lo que si se desea hacer girar el motor en sentido horario a la mitad del par, osea, 0 Nm o 0 cuentas.

ParFinal=256*0+0 = 0

Por lo que si se desea hacer girar el motor en sentido anti-horario a la mitad del par, osea, 0 Nm o 1024 cuentas aproximadamente.

ParFinal=256*4+0=1024

Por lo que si se desea hacer girar el motor en sentido horario a la mitad del par, osea, 0.75 Nm o 512 cuentas.

ParFinal=256*2+0 = 512

Por lo que si se desea hacer girar el motor en sentido anti-horario a la mitad del par, osea,0.75 Nm o 1536 cuentas aproximadamente.

ParFinal=256*6+0=1536
Por lo que si se desea hacer girar el motor en sentido horario a la mitad del par, osea, 1.5 Nm o 1023 cuentas.

ParFinal=256*3+255 = 512

Por lo que si se desea hacer girar el motor en sentido anti-horario a la mitad del par, ósea,
1.5 Nm o 2047 cuentas aproximadamente.

ParFinal=256*7+255=1536

Apéndice C

Ejemplos de Instrucciones del Servomotor Dynamixel AX-12 y AX-18

En este apartado se explicarán las instrucciones que se ocuparon en este trabajo mediante el uso de ejemplos en código hexadecimal. En código hexadecimal se toman 8 bits, por lo que en este caso se envían los datos en decimal tomando desde 0 a 255, para el caso de código hexadecimal es 0x00 hasta 0xFF.

Para el caso de begin como se observa en la librería para Arduino, este corresponde a generar un puerto de comunicación para mantener la

comunicación entre el servomotor y el microcontrolador, por lo que esto no se explicará.

1. **Lectura de datos**

En el trabajo se utiliza la lectura de datos para adquirir los datos de posición del servomotor.

Por lo que si se desea obtener la posición a la que se encuentra el servomotor con ID=1, se debe envíar la siguiente Instrucción:

Paquete de Instrucción:

[255,255,ID,Length,Instruccion,PresentPosition_L,Byte_Read_Position,Checksum]

Parámetros:

Length= 4, esto corresponde a que hay 4 parámetros independientes en la cadena a enviar que son ID, Instrucción, PresentPosition_L y ByteReadPosition.

ID=1, corresponde al número del servomotor que se desea conocer.

Instrucción= 2, corresponde al tipo de dato, el fabricante define un 2 si el dato que se ejecuta es de lectura.

PresentPosition_L= 36, Este corresponde al valor en estado bajo que se desea enviar con- tenido en la tabla de control.

Byte_Read_Position= 2, hace referencia al numero de datos en el que se desea recibir la posición, por lo que para el servomotor se necesita un estado de PosicionN_L y una Po- sicionN_H, un estado bajo y un alto como se observa en readPosition( ) de la librería de Arduino.

Checksum=(OR(ID+ Length+ Instruction+ PresentPosition_L+ ByteReadPosition)) AND (255)

La respuesta del servomotor es la siguiente:

Paquete de Status:

[255,255,ID,Length, Error, PositionN_L, PositionN_H,Checksum]

Del Paquete de Status los que son necesarios de adquirir es PosicionN_L y PosicionN_H, como ya se vio anteriormente, la posición Final del motor esta representada como:

PosicionFinal= 256*PosicionN_H+PosicionN_L

2. **Escritura de datos**

En el trabajo se utiliza la escritura de datos se utiliza para que el servomotor se configure o se desplace hasta algún valor deseado. Al final de cada instrucción el servomotor regresa un Status para verificar si la instrucción se realizó adecuadamente, esto no se mostrará en la explicación.

Por lo que si se desea configurarle el Endless en ON a un ID=2, se debe envíar la siguiente Instrucción:

Paquete de Instrucción:

[255,255,ID,Length, Instruccion, CCW_AngleLimit_L, CCW_AL_L, CCW_AL_H, Check- sum]

Parámetros

Length= 5, esto corresponde a que hay 5 parámetros independientes en la cadena a enviar que son ID, Instrucción, CCW_AngleLimit_L, CCW_AL_L y CCW_AL_H.

ID=2, corresponde al número del servomotor que se desea configurar

Instrucción= 3, corresponde al tipo de dato, el fabricante define un 3 si el dato que se ejecuta es de escritura.

CCW_AngleLimit_L= 8, Este corresponde al valor en estado bajo que se desea enviar con- tenido en la tabla de control.

CCW_AL_L=0 y CCW_AL_H=0

Para CCW_AL_L y CCW_AL_H el fabricante toma valores de 0 para un rotación com- pleta, si se hubiera deseado que el Endless estuviera en OFF, los valores tendrían que ser CCW_AL_L=255 y CCW_AL_H= 3, debido a que CCW_Total = 256*CCW_AL_H+ CCW_AL_L,= 1023, corresponde a la máxima resolución asignada.

Checksum=(OR(ID+ Length+ Instruction+ CCW_AngleLimit_L+ CCW_AL_L+ CCW_AL_H)) AND (255)

Para el TorqueStatus, Si se desea deshabilitar el torque en el servomotor 3. Paquete de Instrucción:
[255,255,ID,Length, Instruccion, TorqueEnable, Status, Checksum]
Parámetros:
Length= 4, esto corresponde a que hay 4 parámetros independientes en la cadena a enviar que son ID, Instrucción, TorqueEnable y Status.

ID=3, corresponde al numero del servomotor que se desea configurar.

Instrucción= 3, corresponde al tipo de dato, el fabricante define un 3 si el dato que se ejecuta es de escritura.

TorqueEnable=24, Este corresponde al valor que se desea enviar contenido en la tabla de control.

Status= 0, El valor del Status corresponde a 1 si se desea habilitar el torque y 0 si se desea deshabilitarlo.

Checksum=(OR(ID+ Length+ Instruction+ TorqueEnable+ Status)) AND (255)

3. **SyncWrite**

En la tesis se utiliza la función de SyncWrite para que todos los motores realicen la misma instrucción, pero se pueden tomar distintos parámetros para cada uno de ellos.

Por lo que si se desea mover los 3 servomotores a una Posicion y velocidad deseada, Si su- ponemos lo siguiente:
Si queremos que el Servomotor 1, se mueva a una velocidad 200 cuentas, y llegue a 512 cuentas o 150 grados.
Si queremos que el Servomotor 2, se mueva a una velocidad 512 cuentas, y llegue a 1023 cuentas o 300 grados.
Si queremos que el Servomotor 3, se mueva a una velocidad 800 cuentas, y llegue a 102 cuentas o 30 grados.

Nota: si el EndLess se encuentra en ON los valores de velocidad corresponderán al par de giro por lo que se deben de tomar de 0-1023 en sentido anti-horario y de 1024-2047. Si el

EndLess está en OFF la velocidad toma de 0-1023.
Paquete de Instrucción:
[255,255,Broadcast_ID,Length, Instruction, StartingAddress, LengthData, ID_1, Position1_L, Position1_H, Speed1_L, Speed1_H, ID_2, Position2_L, Position2_H, Speed2_L, Speed2_H, ID_3, Position3_L, Position3_H, Speed3_L, Speed3_H, Checksum]
Parámetros:
Broadcast_ID=254, esta instrucción permite mandar la instrucción a todos los servomotores, y cada servomotor adquiere de la cadena el dato que le corresponde.

LengthData=4, es el número de datos independientes a mandar a cada motor, son PositionN_L, PositionN_H, SpeedN_L y SpeedN_H.

Length=19, este dato se calcula como: Length=(LengthData+1)*NUMBEROFDYNAMIXEL + 4; en donde NumberOfDynamixel =3, ya que corresponde al número de servomotores al que se le mandara los datos.

StartingAddress=30, este dato se toma como predeterminado para el inicio en modo Syncwrite.

ID_N=N, este corresponde al numero de servomotor que se desea configurar el parámetro. Por lo que en el ejemplo ID_1=1, ID_2=2, y ID_3=3.

PositionN_L , corresponde a la posición en estado bajo de cada servomotor como se muestra en cada servomotor. En el Arduino se representa el comando como PositionN_L= Position-Final (donde toma los 8 bits menos significativos).

Position1_L= 0, Position2_L=255 y Position3_L= 102

PositionN_H, corresponde a la posición en estado alto de cada servomotor como se muestra en cada servomotor. En el Arduino se representa el comando como PositionN_H= Position-Final >> 8 (Toma en cuenta los 8 bits más significativos).

Position1_H= 2, Position2_H=3 y Position3_H= 0

SpeedN_L, corresponde a la velocidad en estado bajo de cada servomotor como se muestra en cada servomotor. En el Arduino se representa el comando como SpeedN_L= SpeedFinal (donde toma los 8 bits menos significativos).

Speed1_L= 200, Speed2_L=0 y Speed3_L= 32

SpeedN_H, corresponde a la velocidad en estado alto de cada servomotor como se muestra en cada servomotor. En el Arduino se representa el comando como SpeedN_H=SpeedFinal >> 8 (Toma en cuenta los 8 bits mas significativos).

Speed1_H= 0, Speed2_H=2 y Speed3_H= 3

Checksum es la suma de todos los valores que se ingresan a la cadena con unas ciertas operaciones AND y OR.