

UNIVERSIDAD POLITÉCNICA
DE LA ZONA METROPOLITANA DE GUADALAJARA



UNIVERSIDAD POLITÉCNICA
DE LA ZONA METROPOLITANA DE GUADALAJARA

Universidad Politécnica de la Zona Metropolitana de Guadalajara.

Brazo Cartesiano.

Integrantes:

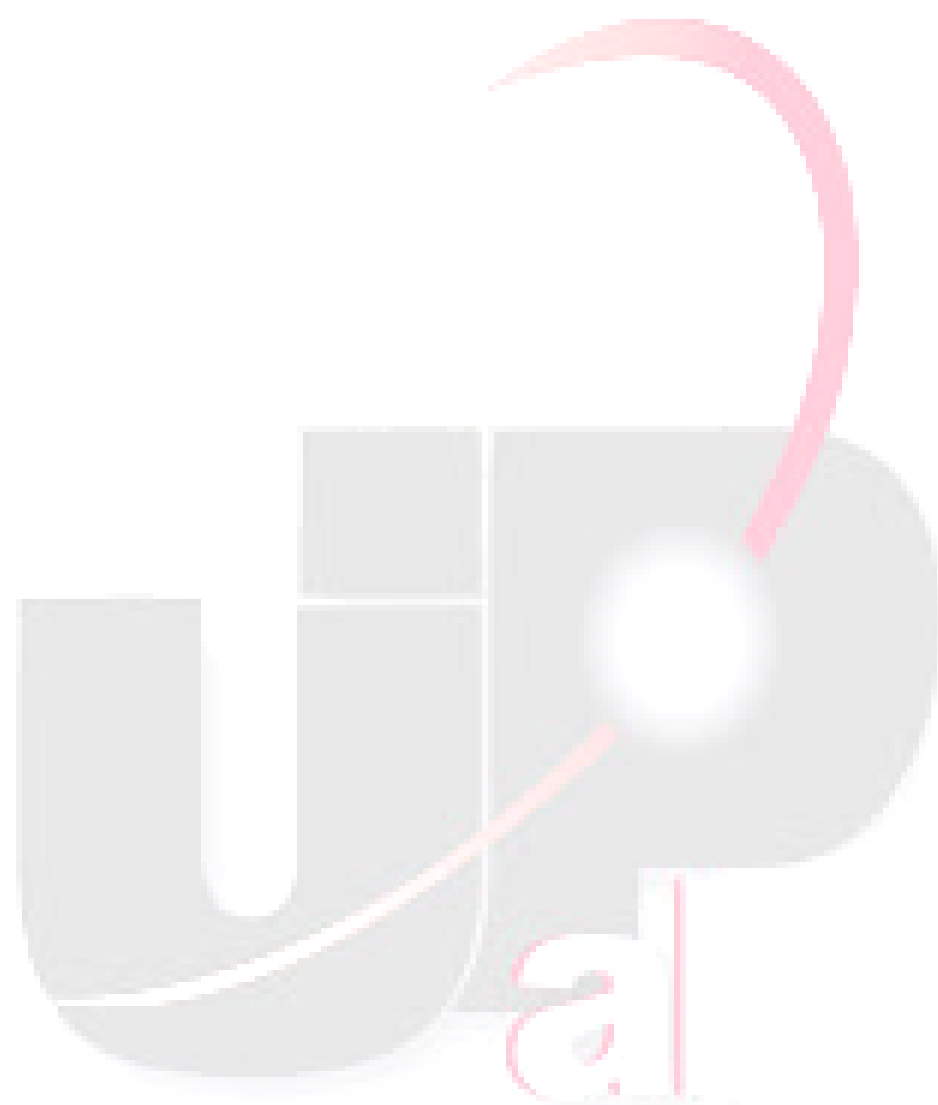
Hernández Castillo Ana Yuritzi.

Hernández García Andrés de Jesús.

Rodríguez Rodríguez José Luis.

Rosales Ortiz Ian Alexis.

UNIVERSIDAD POLITÉCNICA
DE LA ZONA METROPOLITANA DE GUADALAJARA
Tlajomulco de Zúñiga, Jalisco
Agosto 2019



UNIVERSIDAD POLITÉCNICA
DE LA ZONA METROPOLITANA DE GUADALAJARA

Índice:

1.- Introducción.....	7
1.1.- Robótica industrial.....	7
1.2.- Estructura de los robots industriales.....	7
1.3.- Clasificación de los diferentes tipos de robots.....	9
1.4.- Morfología de los robots industriales.....	10
1.4.1.- Configuración Cartesiana.....	10
1.4.2.- Configuración Cilíndrica.....	10
1.4.3.- Configuración Polar.....	10
1.4.4.- Configuración SCARA.....	11
1.4.5.- Configuración Angular.....	11
2.- Planeación.....	12
3.- Justificación.....	14
4.- Objetivo General.....	15
4.1.- Objetivos Específicos:.....	15
5.- Marco Teórico.....	16
5.1.- Historia de la automatización industrial.....	16
5.2.- Robótica industrial.....	17
5.3.- Clasificación del Robot Industrial.....	17
5.4.- Parámetros característicos de los robots industriales.....	19
5.4.1.- Número de grados de libertad	19
5.4.2.- Espacio de accesibilidad o espacio (volumen) de trabajo.....	20
5.4.3.- Capacidad de posicionamiento del punto terminal.....	20
5.4.4.- Capacidad de carga.....	20
5.5.- Configuración cinemática cartesiana de un robot PPP.....	20
5.6.- Robot cartesiano.....	21
5.7.- Funcionalidad de los robots industriales.....	22
6.- Diseño mecánico.....	23
6.1.- Base eje “Y”.....	23

6.2.-	Puente eje “X”.....	23
6.3.-	Cabecal eje “Z”.....	24
6.4.-	Ensamble completo.....	25
6.5.-	Cálculo del motor de traslación.....	26
6.6.-	Resistencia de la rodadura.....	27
7.-	Diseño Electrónico.....	29
7.1.-	Alimentación.....	29
7.2.-	Motores a pasos.....	29
7.3.-	Drivers.....	30
7.4.-	CNC Shield.....	31
8.-	Diseño del control.....	33
8.1.-	Microcontrolador.....	33
8.2.-	Programador.....	33
8.2.1.-	Desarrollo de aplicaciones en Mbed.	33
8.2.2.-	Mbed OS	34
8.3.-	ROS.....	34
8.4.-	Código de programación.....	35
8.4.1.-	Comandos desde ROS.....	36
9.-	Conclusiones.....	38
9.1.-	Hernández Castillo Ana Yuritzi.....	38
9.2.-	Hernández García Andrés de Jesús.....	38
9.3.-	Rodríguez Rodríguez José Luis.....	38
9.4.-	Rosales Ortiz Ian Alexis.....	39
10.-	Bibliografía.....	40
Anexos A.	Instalando ROS en Ubuntu.....	41

UNIVERSIDAD POLITÉCNICA
DE LA ZONA METROPOLITANA DE GUADALAJARA

Índice de imágenes:

Imagen 1. Analogía: Brazo Humano – Brazo Robot.....	7
Imagen 2. Elementos estructurales de un robot industrial.....	8
Imagen 3. Tipos de articulaciones de un robot: a) lineal, b) rotacionales	8
Imagen 4. Punto terminal de un manipulador.....	9
Imagen 5. Configuración cartesiana con su volumen de trabajo.....	10
Imagen 6. Configuración cilíndrica con su volumen de trabajo.....	10
Imagen 7. Configuración polar con su volumen de trabajo.....	11
Imagen 8. Configuración Angular.....	11
Imagen 9. Configuración del robot cartesiano.....	21
Imagen 10. Base eje “Y”.....	23
Imagen 11. Placas de Acero eje “X”.....	24
Imagen 12. Eje “Y”.....	25
Imagen 13. Ensamble Completo.....	26
Imagen 14. Partes del motor a pasos.....	30
Imagen 15. Conexión del driver.....	31
Imagen 16. CNC Shield.....	32
Imagen 17. Pines de Freescale KL25Z.....	33
Imagen 18. Roscore.....	36
Imagen 19. Puerto serial.....	36
Imagen 20. Ejecutando roscore.....	41

UNIVERSIDAD POLITÉCNICA
DE LA ZONA METROPOLITANA DE GUADALAJARA

1. Introducción.

1.1. Robótica industrial.

A lo largo de la historia el ser humano ha creado herramientas y artefactos que lo ayuden a realizar tareas que tal vez para una sola persona resulte complicado y/o peligroso; y así el hombre llegó a realizar grandes inventos de máquinas que tienen varias ventajas entre ellas: maximizar costos, reducir riesgos para obreros, disminuir tiempos de operación, producción en serie etc.

En la actualidad los robots juegan un papel fundamental en el desarrollo de empresas y en la creación de nuevas tecnológicas. El robot industrial es la unión de una parte mecánica con una parte electrónica que a su vez funciona como dispositivo de control principal. (Antonio, Peñín, Balaguer, & Aracil, 2019)

A los manipuladores robóticos se les suele denominar también brazos de robot por la analogía que se puede establecer, en muchos casos, con las extremidades superiores del cuerpo humano (Imagen 1).

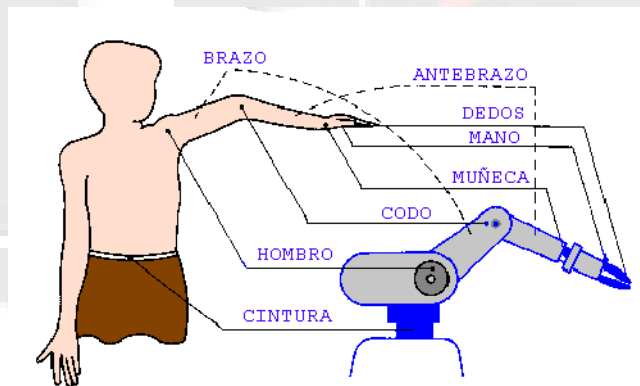


Imagen 1. Analogía: Brazo Humano – Brazo Robot

1.2. Estructura de los robots industriales.

Un manipulador robótico consta de una secuencia de elementos estructurales rígidos, denominados enlaces o eslabones, conectados entre sí mediante juntas o articulaciones (imagen 2), que permiten el movimiento relativo de cada dos eslabones consecutivos. (Alonso, s.f.)

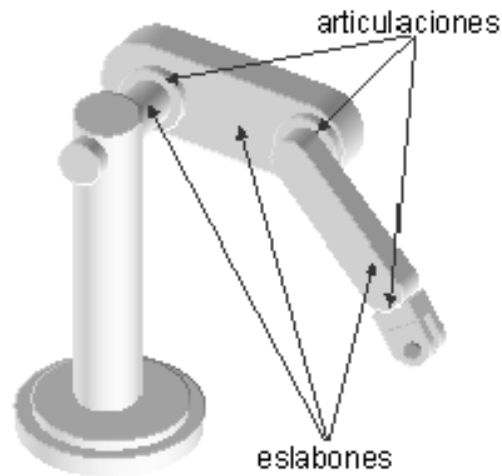


Imagen 2. Elementos estructurales de un robot industrial

Una articulación puede ser:

- Lineal (deslizante, traslacional o prismática), si un eslabón desliza sobre un eje solidario al eslabón anterior (a) Imagen 3).
- Rotacional, en caso de que un eslabón gire en torno a un eje solidario al eslabón anterior (b) Imagen 3).

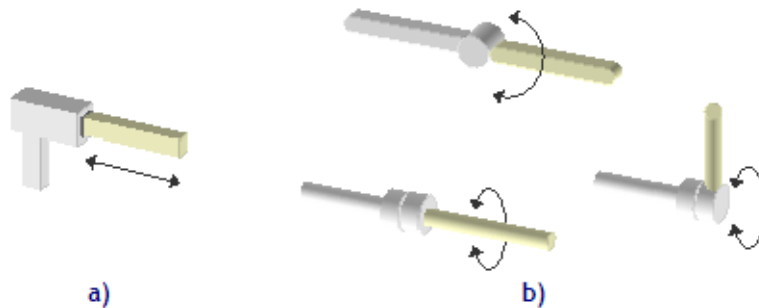


Imagen 3. Tipos de articulaciones de un robot: a) lineal, b) rotacionales

El conjunto de eslabones y articulaciones se denomina cadena cinemática. Se dice que una cadena cinemática es abierta si cada eslabón se conecta mediante articulaciones exclusivamente al anterior y al siguiente, exceptuando el primero, que se suele fijar a un soporte, y el último, cuyo extremo final queda libre. (Renteria & Rivas) A éste se puede conectar un elemento terminal o actuador final: una herramienta especial que permite al robot de uso general realizar una aplicación particular, que debe diseñarse específicamente para dicha aplicación: una herramienta de sujeción, de soldadura, de pintura, etc. El punto

más significativo del elemento terminal se denomina punto terminal (PT). En el caso de una pinza, el punto terminal vendría a ser el centro de sujeción de la misma (Imagen 4).

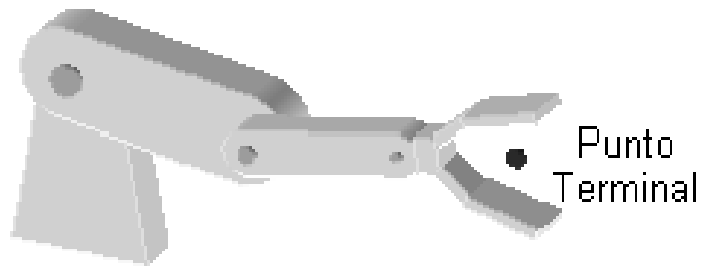


Imagen 4. Punto terminal de un manipulador

Se denomina grado de libertad (g.d.l.) a cada una de las coordenadas independientes que son necesarias para describir el estado del sistema mecánico del robot (posición y orientación en el espacio de sus elementos). Normalmente, en cadenas cinemáticas abiertas, cada par eslabón-articulación tiene un solo grado de libertad, ya sea de rotación o de traslación. Pero una articulación podría tener dos o más grados de libertad que operan sobre ejes que se cortan entre sí. (Reyes Cortés)

1.3. Clasificación de los diferentes tipos de robots.

1. Tipo A. Manipuladores: son robots multifuncionales con sistemas mecánicos básicos, por lo que deben ser utilizados en tareas sencillas y repetitivas, pueden ser controlados por una o más personas mediante control remoto, y cuenta con tres o cuatro grados de libertad.

2. Tipo B. Computarizados de precisión por controlador lógico programable (PLC): es un manipulador pre ajustable que cuenta con sensores de regulación, por lo tanto tienen mayor precisión y fuerza, se regulan mediante un PLC y tiene 4 grados de libertad.

3. Tipo C. Computarizados por CNC: robot programable con trayectoria continua, son equipos más avanzados programados por CNC, con mayor fuerza y realizan trabajos más exigentes, posee seis grados de libertad.

4. Tipo D. Sensoriales: robot que mediante sensores adquiere información de su entorno y es capaz de adaptarse a las condiciones del mismo. Poseen seis grados de libertad con una precisión de $\pm 0,04$ mm. (Renteria & Rivas)

1.4. Morfología de los robots industriales.

1.4.1 Configuración Cartesiana

Posee tres grados de libertad con movimientos lineales, los cuales corresponden a los tres ejes X, Y y Z. Los movimientos que realiza este tipo de robot los hace mediante interpolaciones lineales. Presenta volúmenes de trabajo regulares.

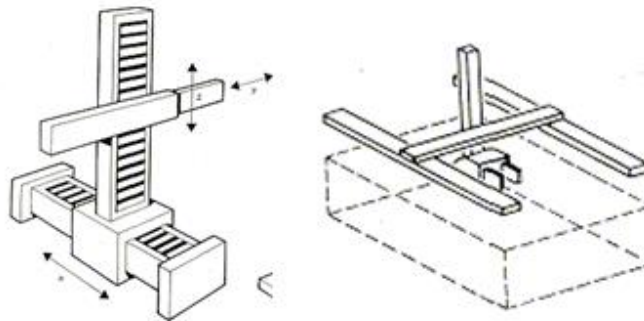


Imagen 5. Configuración cartesiana con su volumen de trabajo.

1.4.2 Configuración Cilíndrica

Posee tres grados de libertad con dos movimientos lineales y dos de rotación, por lo tanto, los movimientos se basan en interpolación lineal e interpolación por articulación. Presenta un volumen de trabajo parecido a un cilindro.

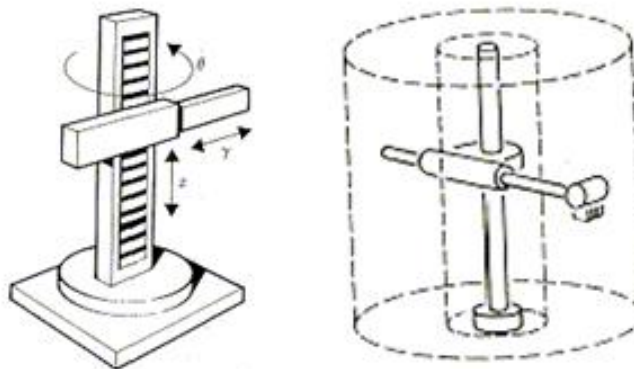


Imagen 6. Configuración cilíndrica con su volumen de trabajo.

1.4.3 Configuración Polar

Posee tres grados de libertad con movimientos de rotación y uno lineal, utiliza la interpolación por articulación para sus dos primeros movimientos y la lineal para movimientos de extensión. Volumen de trabajo irregular.

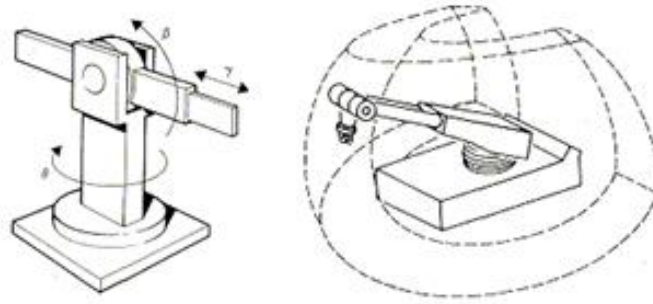


Imagen 7. Configuración polar con su volumen de trabajo.

1.4.4 Configuración SCARA

Posee varios tipos de articulaciones, combinaciones de las anteriores. Es destacable la configuración SCARA (Selective Compliance Assembly Robot Arm).

1.4.5 Configuración Angular

Posee una articulación con movimientos rotacionales y dos angulares. El movimiento se basa en interpolación por articulación. El volumen de trabajo es irregular, por lo que suele revisar el plano del robot

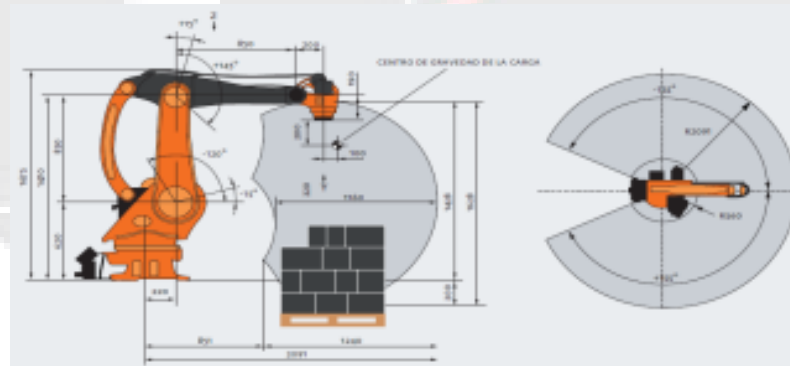
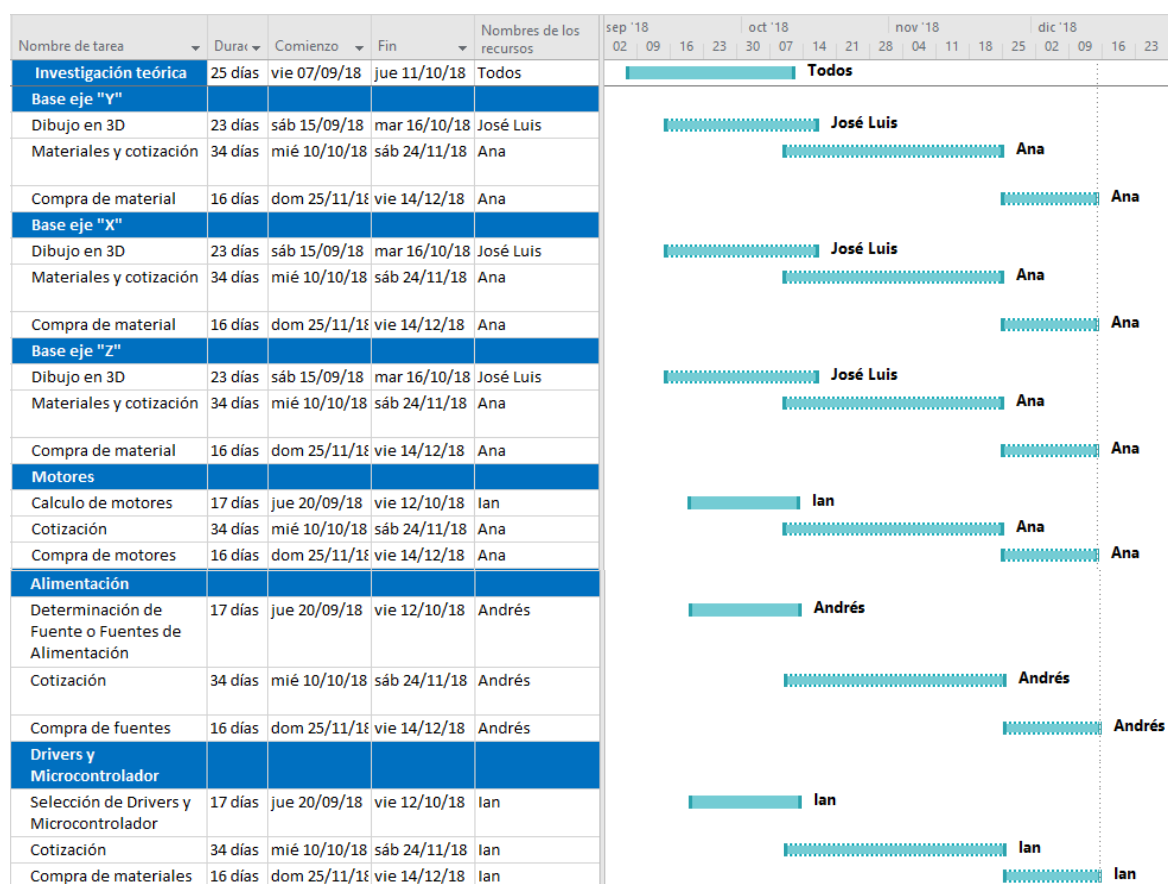


Imagen 8. Configuración Angular.

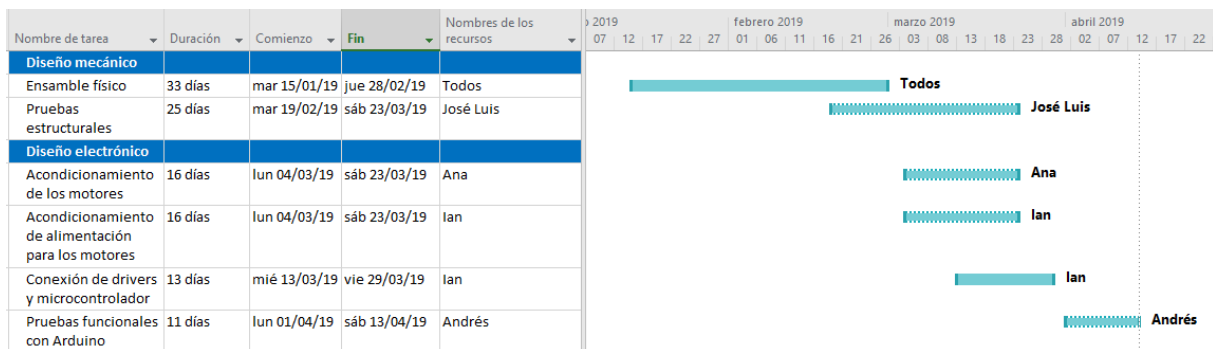
2. Planeación.



La planeación se encuentra dividida en tres bloques de cuatro meses, el primer bloque que se muestra en la ilustración anterior corresponde a la gestión de tiempos y tareas, es decir la definición de todas las etapas del proyecto (etapa mecánica, etapa electrónica y etapa de control) y posteriormente el desglose de todas las actividades necesarias para la culminación de cada etapa.

La importancia de esta etapa recae en el hecho de designar encargados por cada actividad, el primer bloque tiene como principal objetivo realizar toda la documentación necesaria para su elaboración, simulaciones, cálculos estructurales y selección de componentes para obtener los presupuestos y proceder a la compra de los materiales.

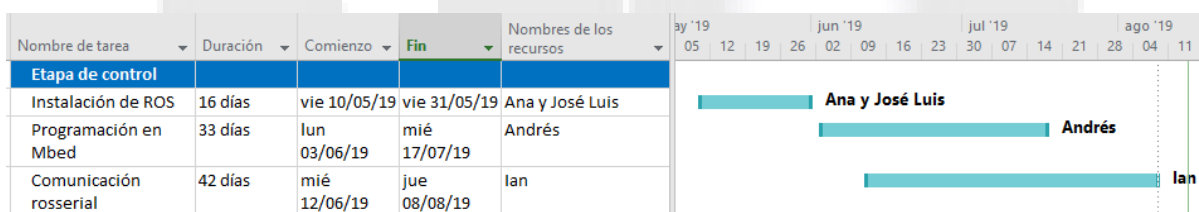
UNIVERSIDAD POLITÉCNICA
DE LA ZONA METROPOLITANA DE GUADALAJARA



La etapa dos se concentra en la parte mecánica y electrónica, después de haber concluido la primera etapa, es decir ya contando con los materiales necesario, se procederá a ensamblar cada eje por individual, una vez terminado se ensamblará el brazo completo.

Al terminar el ensamble se instalarán los motores y la alimentación, se realizarán pruebas de la estructura, para ver si los motores tienen la fuerza para deslizar cada eje.

Se realizará un pequeño programa en Arduino para determinar si las conexiones de los motores son correctas.



La tercera etapa se concentra únicamente en la programación del microcontrolador y la parte esencial de este proyecto la comunicación con ROS, la comunicación entre el sistema operativo y el microcontrolador para poder darle indicaciones desde nuestro ordenador.

3. Justificación.

La automatización se compone de todas las teorías y tecnologías en caminadas a sustituir el trabajo del hombre por una máquina, este flujo de trabajo resulta inevitable en la actualidad, ya que los proyectos de ingeniería han alcanzado una complejidad tal que es inconcebible realizarlos sin antes disponer de la seguridad de un diseño en ordenador, junto con simulaciones y todos los parámetros que aseguren su correcto funcionamiento. Actualmente, la tecnología está ampliamente extendida, dejando cada fuera de lugar a los métodos artesanos.

Inicialmente, el factor predominante que condicionó todo automatismo fue el aumento de la productividad. Posteriormente, debido a las nuevas necesidades de la industria aparecieron otros factores no menos importantes como la precisión, la rapidez y la flexibilidad.

La automatización industrial surge y se desarrolla a lo largo del tiempo por la exigencia de cubrir ciertas necesidades: como la de fabricar productos que no se podían conseguir en cantidad y calidad suficientes sin recurrir a la automatización del proceso de fabricación, los escasos de productos difíciles de fabricar, por ser excesivamente complejos para ser controlados por un operador humano, la necesidad de fabricar productos con unos costes de producción bajos, etc.

4. Objetivo General.

Construir un prototipo industrial CNC cartesiano de 3GDL.

4.1. Objetivos Específicos:

- Diseñar estructura mecánica para el robot cartesiano.
- Diseñar sistema de transmisión.
- Diseñar sistema de soporte y deslizamiento.
- Desarrollar cálculos estructurales, de funcionamiento de la propuesta seleccionada.
- Acondicionar cableado y alimentación.
- Programar motores.
- Elaborar un prototipo de bajo coste.
- Utilizar plataformas de libre acceso y fáciles de manejar.

5. Marco Teórico.

5.1. Historia de la automatización industrial

Desde muchos años antes de que existiera la tecnología necesaria para la creación de robots, la humanidad ya soñaba con estos, prueba de ello fueron los escritos del visionario inventor Nikola Tesla, quien desde 1890 imaginaba máquinas que eran capaces de reemplazar a los humanos en varias tareas.

Pero la palabra robot fue popularizada por el escritor Karel Capek, quien en 1921 estrenó su obra Rossum's Universal Robots, también conocida como R.U.R., el término tiene su raíz en el vocablo checo “Robota” cuya traducción es trabajo forzado, esclavo o servicio; en esta obra de ficción se emplea la expresión para designar a las máquinas que trabajan al servicio del hombre.

Fue George Devol, pionero en la robótica industrial, quien elaboró el primer robot funcional en los términos que se conocen actualmente, en 1954 utilizó patentes electrónicas de inventos que había hecho con anterioridad para crear a Unimate, un dispositivo multifuncional que podía ser empleado en distintas tareas. En 1962 la primera de estas máquinas fue instalada en la línea de producción de General Motors, con la finalidad de ensamblar motores, convirtiéndose en la primera cadena de producción automatizada de la historia.

A raíz del éxito que tuvo el Unimate, inventores de todo el mundo comenzaron a desarrollar tecnologías que permitieran mejorar la funcionalidad de los robots industriales, y su uso se extendió en todos los campos posibles: industrial, militar, espacial e incluso nuclear, en especial con la llegada de los autómatas a países como Japón, que es hoy pionero mundial en el perfeccionamiento de estos artilugios.

La carrera espacial y la implementación de microprocesadores tuvieron gran influencia en el progreso del campo robótico, pues la necesidad de precisión al manejar ámbitos tan complicados animó a los científicos a crear mejores autómatas con grandes destrezas y mayor exactitud.

El año 1980 es conocido como el primero de la “Era robótica”, pues la producción de estas máquinas aumentó un 80% en comparación con el año anterior y para 1987 se fundó la

Federación Internacional de Robótica (IFR por sus siglas en inglés), cuyo propósito es promover la investigación, el desarrollo, el uso y la cooperación internacional de la robótica.

5.2. Robótica industrial.

En la actualidad los robots manipuladores son considerados como los robots más útiles dentro de la rama industrial. Para lograr una definición clara y concisa de lo que significa es necesario entender cada una de las ideas que cada Asociación impone.

La definición comúnmente aceptada es la de la Asociación de Industrias de Robótica (RIA, Robotic Industry Asociación) la cual lo define como: "Un robot industrial es un manipulador multifuncional reprogramable, capaz de mover materias, piezas, herramientas, o dispositivos especiales, según trayectorias variables, programadas para realizar tareas diversas".

También se incluye la definición establecida por la Asociación Francesa de Normalización (AFNOR), la cual ve necesario definir primero que es un manipulador y en base a esta la del robot, definiendo entonces un manipulador como un mecanismo formado por una sucesión de elementos en serie, articulados entre si y destinados a la sujeción y traslado de objetos.

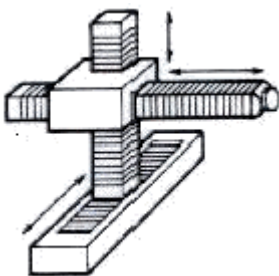
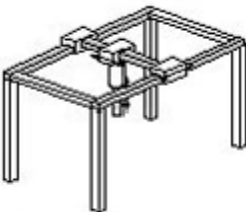
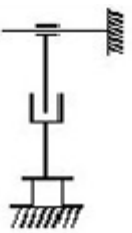

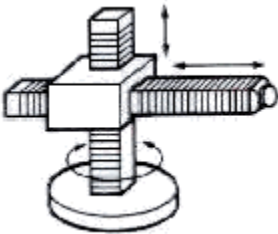







Un robot industrial puede ser manipulado directamente por un operador humano o mediante un control lógico computacional y un robot definido como un manipulador automático controlado, reprogramable, polivalente, capaz de posicionar y orientar elementos siguiendo una trayectoria fija o programable.

El robot se compone de uno o varios eslabones que terminan en una muñeca. Poseen en su unidad de control un banco de memoria y puede también ser perceptor del entorno en el que trabaja mediante el uso de elementos sensoriales, cumplen funciones de manera cíclica y se adaptan a otras sin cambios en su diseño físico ni de material. (Cazar, 2008)

5.3. Clasificación del Robot Industrial.

Con el surgimiento de microcontroladores y a la implementación de motores a pasos en lazos cerrados y servomotores, que permiten conocer con exactitud la posición real de los componentes de un robot y en el establecimiento del error con la posición deseada, se dio

origen a una serie de tipos de robots, uno de estos son los robots de repetición o aprendizajes los cuales mediante el uso de controladores manuales o dispositivos auxiliares, repiten una secuencia de movimientos previamente realizados.

Configuración geométrica	Estructura cinemática	Espacio de trabajo	Ejemplo
<p>cartesianos</p>  <p>tipo antilével</p>  <p>tipo pórtico</p>			
<p>cilíndrico</p> 			
<p>polar</p> 			

DE LA ZONA METROPOLITANA DE GUADALAJARA

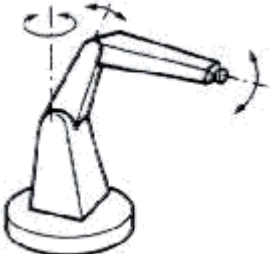
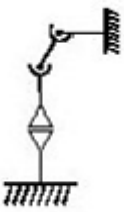



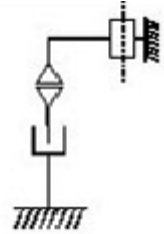



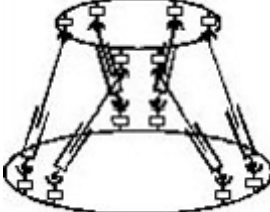


<p>esférico</p> 			
<p>SCARA</p> 			
<p>paralelo</p> 			

Tabla 1. Configuraciones Geométricas, Estructura Cinemática, Espacio de Accesibilidad y Ejemplos de Robots Industriales

Otros son conocidos como robots inteligentes son capaces de relacionar y tomar decisiones dependiendo del medio que los rodea mediante la utilización de sensores, además existen los robots controlados por computador cuando además de ser controlados por un microcontrolador se dispone de un lenguaje compuesto por instrucciones adaptadas para el tipo de robot, con las que se puede ejecutar un programa secuencial para la elaboración de un proceso. A este tipo de programación se le llama textual y no depende para su creación del manipulador. (Caparros & Hernández, 1999)

5.4. Parámetros característicos de los robots industriales.

5.4.1. Número de grados de libertad

Es el número total de grados de libertad de un robot, dado por la suma de g.d.l. de las articulaciones que lo componen. Aunque la mayoría de las aplicaciones industriales requieren

6 g.d.l., como las de soldadura, mecanizado y almacenamiento, otras más complejas requieren un número mayor, tal es el caso de las labores de montaje.

5.4.2. Espacio de accesibilidad o espacio (volumen) de trabajo

Es el conjunto de puntos del espacio accesibles al punto terminal, que depende de la configuración geométrica del manipulador. Un punto del espacio se dice totalmente accesible si el PT puede situarse en él en todas las orientaciones que permita la constitución del manipulador y se dice parcialmente accesible si es accesible por el PT pero no en todas las orientaciones posibles.

5.4.3. Capacidad de posicionamiento del punto terminal

Se concreta en tres magnitudes fundamentales: resolución espacial, precisión y repetibilidad, que miden el grado de exactitud en la realización de los movimientos de un manipulador al realizar una tarea programada.

5.4.4. Capacidad de carga

Es el peso que puede transportar el elemento terminal del manipulador. Es una de las características que más se tienen en cuenta en la selección de un robot dependiendo de la tarea a la que se destine.

5.5. Configuración cinemática cartesiana de un robot PPP.

Es una configuración compuesta por tres articulaciones (3D o PPP). La posición de cada una de las articulaciones es controlada mediante coordenadas cartesianas (" x " y " y ", si es de dos ejes o " x ", " y " y " z ", si es de tres).

Gracias a que los movimientos pueden iniciarse o detenerse simultáneamente en sus tres ejes el movimiento de la herramienta es mucho más suave, de igual forma permite que el robot se mueva directamente a una referencia, en lugar de seguir trayectorias paralelas a cada eje, una de las ventajas de un robot cartesiano es que sus movimientos son totalmente lineales permitiendo así la implementación de controles más simples, de igual manera tienen un alto grado de rigidez mecánica, precisión y repetibilidad, pueden llevar cargas pesadas a lo largo de su campo de trabajo.

En cuanto a sus desventajas los robots cartesianos son generalmente limitados en sus movimientos a su espacio de trabajo. La aplicación para la que se va a utilizar robot cartesiano en este proyecto consiste en la instalación de una herramienta de corte en la muñeca del robot con el fin de lograr mecanizados didácticos en materiales de bajos esfuerzos cortantes. Basado en el control numérico computarizado y en el funcionamiento de las fresadoras CNC. (Rojas, Mahla, Muñoz, & Castro, 2003)

5.6. Robot cartesiano.

Dentro de las aplicaciones de las máquinas automatizadas, se encuentra el robot cartesiano XYZ, es una máquina utilizada en ingeniería para tareas de soldadura, mecanizado, pintura y la fabricación de diversos elementos mediante el desprendimiento de material generado por una herramienta de corte o por el aporte de material en estado pastoso mediante capas para formar un elemento completo.

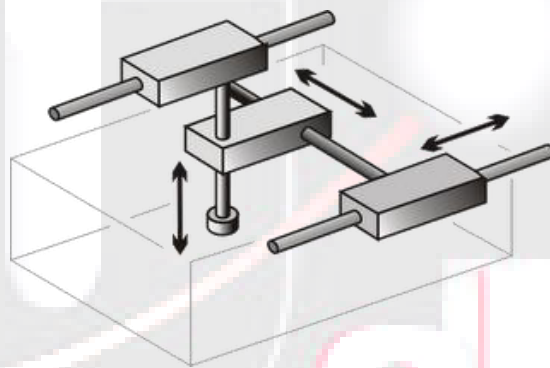


Imagen 9. Configuración del robot cartesiano

El robot cartesiano consiste en la combinación de movimientos lineales independientes que generan trayectorias complejas en un espacio tridimensional.

Un robot cartesiano es una configuración de tres articulaciones prismáticas, cuyas variables son las coordenadas cartesianas de la herramienta utilizada como efector con respecto a la base del mismo, es una configuración simple y sus aplicaciones principalmente se enfocan en campos de ensamble, almacenaje de productos y conformación de objetos mediante la aportación o eliminación de material.

Algunos investigadores han propuesto métodos de configuración de los tres ejes, del mismo modo la forma en que estos se mueven por ejemplo se puede tener el eje Z sin

movimiento y que la mesa de trabajo se mueva en X o Y, la decisión de cómo establecer esta configuración depende netamente de las aplicaciones y requerimientos de funcionalidad de la máquina, adicionalmente se pueden implementar distintos tipos de elementos activos para el control y funcionamiento de cada eslabón del robot, como lo son los motores a paso y los sensores tipo encoders.

Este tipo de máquinas son costosas comercialmente, su infraestructura es cerrada y no permite modificaciones para el desarrollo de prácticas con fines académicos e investigativos. Es por esto que en el presente proyecto se realiza el diseño y construcción de un robot cartesiano que permita estudiar los problemas de diseño, mecanizado y control; de modo que se pueda manipular materiales y herramientas de trabajo liviano, imitando lo que pasaría en un equipo industrial, además de ser un equipo de bajo costo. (Reyes Cortés).

5.7. Funcionalidad de los robots industriales.

Los robots industriales pueden ser utilizados por las compañías para cumplir con una infinidad de tareas, ya que son más precisos, efectivos y rápidos que la mano de obra humana.

Algunos de sus usos regulares son la manipulación de materiales delicados, soldadura, carga, descarga y corte de elementos pesados, aplicación de productos como pintura, baños de metales o ácidos, medición de sólidos, líquido, gases, o aplicación de altas presiones, e incluso pueden funcionar para la supervisión del control de calidad de los materiales fabricados.

El uso de estas maquinarias en diversas industrias ya es algo común y han convertido trabajos difíciles en tareas comunes, reemplazando a los humanos y simplificando líneas de producción complejas, maximizando las ganancias y reduciendo los costos.

Sin embargo, el futuro de los robots aún es incierto y todavía se encuentran lejos de alcanzar su máximo potencial, las industrias no son el único campo interesado en su desarrollo, de hecho, los científicos dedicados a trabajar en la evolución de estos artilugios esperan que en unos cuantos años puedan asemejarse más a los humanos y ser empleados en física, medicina, astronomía y todos los campos de investigación posibles. (Reyes Cortés)

6. Diseño mecánico.

6.1. Base eje “Y”.

El movimiento en el eje “Y” implica básicamente el uso de dos sistemas de deslizamiento en conjunto con un tercero que proporciona el movimiento a lo largo de la dirección lineal. Para poder realizar la tarea de deslizamiento, los sistemas que se seleccionaron tienen solo la función de proporcionar el libre traslado del mecanismo en dicha dirección, a través de unas guías conocidas como ejes de rodamientos lineales.

El primer sistema se encuentra constituido por un rodamiento particular, conocido como rodamiento de transmisión lineal, en conjunto con sus ejes circulares, mientras que el segundo sistema está formado por un rodamiento lineal de bolas que se desplazan a lo largo de perfiles de acero inoxidable conocidos como guías deslizables (Imagen 10).

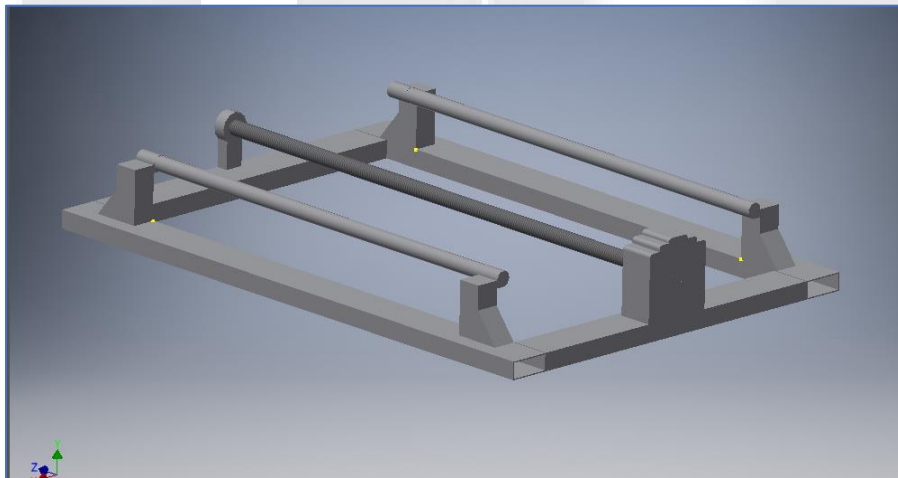


Imagen 10. Base eje “Y”

Los ejes circulares (Imagen 10) están ubicadas a los costados de la mesa y están fijadas a las mismas placas de acero inoxidable que sirven para unir la mesa, el rodamiento lineal de bola está alojado en pequeños cubos de acero maquinados de tal forma que se acoplen a los mismos, permitiendo a los rodamientos trasladarse a lo largo de los ejes circulares.

6.2. Puente eje “X”.

El desplazamiento en la dirección “X” es similar al desplazamiento en la dirección “Y”. El sistema cuenta con una serie de acoplamientos lineales con los que se garantiza el libre

estado en esta dirección, así como también una estabilidad al momento de realizar el maquinado, ya que sobre este mecanismo actúan indirectamente los esfuerzos de corte que son aplicados a la herramienta al encontrarse posicionado de forma perpendicular al corte.

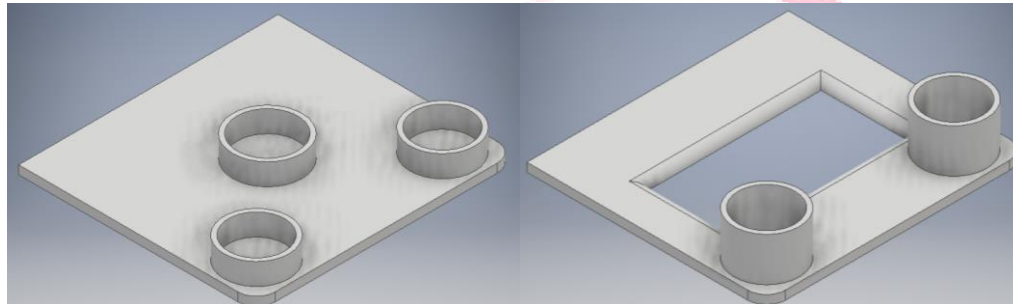


Imagen 11. Placas de Acero eje “X”

El sistema de movimiento en “X” se encuentra unido al sistema de movimiento en “Y” por medio de dos placas de acero maquinadas llamadas “piezas laterales”, estas placas están ensambladas en los rodamientos de transmisión lineal del sistema “Y”, de tal forma que puedan conectar entre sí con ayuda del puente que une cada uno de los extremos del mecanismo y así realizar un traslado en conjunto (Imagen 11).

A cierta altura de estas placas se encuentran ensamblados 2 ejes de acero inoxidable, su función es servir de guías para cada uno de los juegos de rodamientos lineales (dos rodamientos en cada eje), los cuales se deslizan a lo largo de los mismos para poder realizar el traslado del carro. Los rodamientos lineales se encuentran alojados en cubos maquinados de aluminio similares a las cajas de los rodamientos en el sistema en “Y”, estos cubos se encuentran unidos entre sí por una placa maquinada en acero llamada “placa soporte de rodamientos en X”.

6.3. Cabezal eje “Z”.

El sistema de movimiento en dirección “Z” se encuentra ensamblado directamente en el eje “X”, este le proporciona un desplazamiento vertical y es el responsable de llevar a cabo la profundidad de corte en la pieza de trabajo, sin embargo, este sistema no cambia demasiado a comparación de los sistemas anteriores, ya que el principio de funcionamiento es prácticamente el mismo.

El sistema de movimiento en “Z” se une con el sistema de movimiento en “X” por medio de la placa de acero conocida como placa soporte de rodamientos en “X”, ya mencionados en la sección 6.2, une a todos los componentes del sistema de dicha dirección.

En la placa soporte de rodamientos en “X” se ensamblan dos placas de acero, una en la parte superior y una en la parte inferior, las cuales se les dio el nombre de “base superior de movimiento en Z” y “base inferior de movimiento en Z”. Estas uniones se encuentran fijadas con tornillos de tipo allen en los extremos inferior y superior.

Entre las placas base inferior y superior se encuentran colocados dos ejes de acero inoxidable separados a cierta distancia, los cuales funcionan como guías para los rodamientos de transmisión lineal, estos rodamientos lineales se encuentran alojados en cubos maquinados de aluminio que sirven como caja, similares a los que se utilizaron para los sistemas anteriores.

En medio de los dos ejes guías se localiza el tornillo sinfín, el cual se sujeta a las placas superior e inferior por medio de los rodamientos axiales, este ensamble permite que el tornillo sinfín gire de forma libre.

Para que el tornillo sinfín transforme el movimiento giratorio en movimiento lineal, se incorpora un ángulo de aluminio, el cual es atornillado a la placa de soporte, en este pequeño ángulo se encuentra fijado el rodamiento de bolas tipo husillo, el cual realiza el trabajo de desplazar al sistema de forma vertical (Imagen 12).

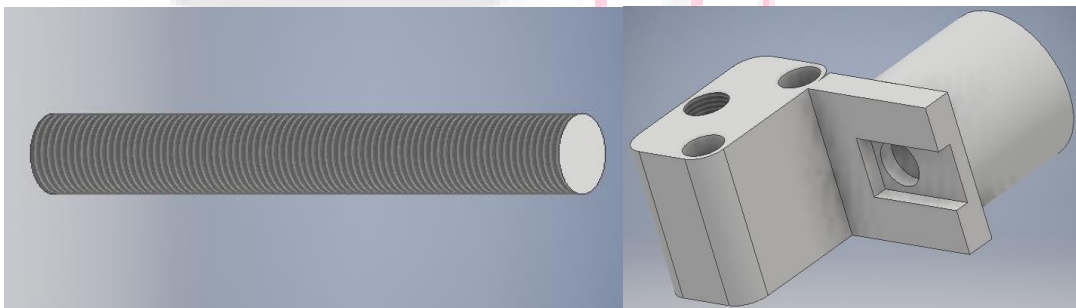


Imagen 12. Eje “Y”

6.4. Ensamble completo.

Una de las características que tiene el diseño de esta estructura es el poder armar y desarmar intuitivamente cada uno de los componentes de la misma, para poder realizar esa

tarea se seleccionaron herramientas comunes, que encontramos normalmente en nuestras casas, sin dejar a un lado la seguridad en las uniones, es por esto que toda la estructura se encuentra ensamblada con tornillos estándar y allen así como también tuercas de seguridad para fijarlos y asegurar el ajuste adecuado para cada pieza.

Dentro de las piezas que conforman la estructura existen piezas maquinadas, las cuales no tienen mayor complicación en su manufactura, sin embargo, estas piezas requieren ser hechas por personal capacitado. Otras piezas son proporcionadas por distribuidores con las características necesarias para ser ensambladas directamente en la máquina, como lo son: los diferentes tipos de rodamientos y los ejes para cada rodamiento.

Finalmente, en la Imagen 13 se muestra el ensamble de los tres sistemas.

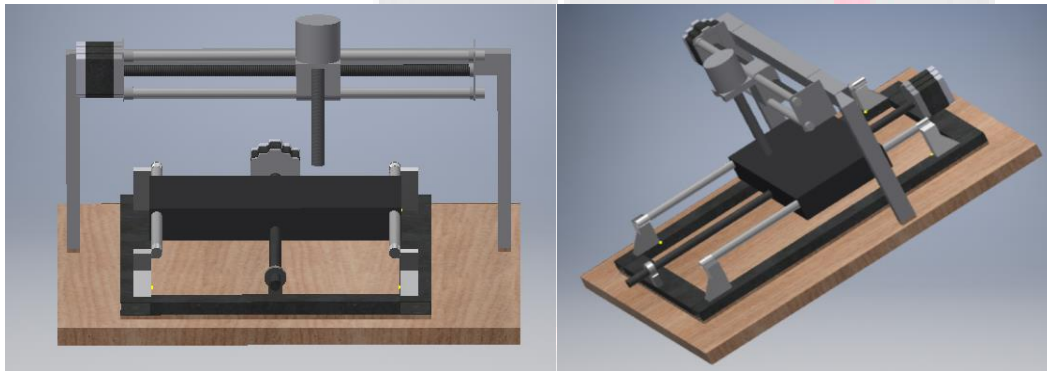


Imagen 13. Ensamble Completo

6.5. Cálculo del motor de traslación.

La velocidad angular de giro en el de la rueda de traslación se calcula conociendo la velocidad máxima del traslado y el diámetro de la rueda (1).

$$Velocidad_{m\grave{a}x} = 3m/s \quad (180m/min) \quad (1)$$

$$D_{rueda} = 500mm$$

La velocidad angular deseada en el eje del motor será la siguiente conociendo la relación del reductor y la velocidad deseada del eje de salida (de traslación (2)).

$$n_{motor} = n_{s \text{ reductor}} \quad i = 114.59 * 19.74 = 2262 \text{ rev/min} \quad (2)$$

Esta es la velocidad angular deseada del motor, pero como se regulará mediante un variador de frecuencia, se deberá seleccionar una frecuencia determinada para obtenerla.

Si conocemos la velocidad del eje motor a 50 Hz, que es de 1460.76 rev/min, podemos conocer la frecuencia (3) que se deberá escoger para obtener la velocidad necesitada, sabiendo que la velocidad del motor es proporcional a la de la frecuencia.

$$\frac{2262 \text{ rev/min}}{1460.76 \text{ rev/min}} = \frac{f(\text{Hz})}{50 \text{ Hz}} \quad (3)$$

$$f = \frac{2262 \text{ rev/min}}{1460.76 \text{ rev/min}} * 50 \text{ Hz} = 77 \text{ Hz}$$

6.6. Resistencia de la rodadura.

La fuerza de fricción (4) que se produce en la rodadura entre la rueda y el carril de acero se calcula como:

$$Fr = M g * \left[\frac{2}{D} * \left(M_r * \frac{d}{2} + f \right) * c \right] \quad (4)$$

Donde:

Fr =Fuerza de fricción por rodadura (N)

M =Masa total de la maquina (Kg)

D =Diámetro de la rueda (mm)

M_r, f, c =Valores para ruedas con rodamientos

$$\therefore F_r = 18800(9.81) * \left[\frac{2}{500} * \left(0.005 * \frac{100}{2} + 0.5 \right) * 0.002 \right]$$

$$Fr=887.8N$$

6.7. Potencia continua en el eje motor.

$$P_x = \frac{F_r V_{max}}{100 * n} \quad (5)$$

$$\therefore P_x = \frac{887.805 * 3}{1000 * 0.95} = 2.80 \text{ kW}$$

De este modo el par resistente en el eje motor en funcionamiento a velocidad constante se calcula conociendo la velocidad del eje motor de 2262 rev/min.

$$M_x = \frac{P_x * 9550}{n_{motor}} \quad (6)$$

Donde:

M_x = Par resistente (Nm)

P_x = Potencia (kW)

n_{mot} = Velocidad de giro del motor (rev/min)

$$\therefore M_x = \frac{2.8 * 9550}{2260} = 11.82 \text{ Nm}$$

7. Diseño Electrónico.

7.1. Alimentación

Para alimentar todo el conjunto se implementó una fuente de alimentación de ordenador de tipo ATX. Esta solución es perfecta ya que tiene precio bajo, cuenta con una de multitud de salidas a diferente tensión y además dispone de ventilador para contribuir a la refrigeración.

Se ha empleado el modelo TMXPT0500 de TECNIMAX, que ofrece hasta una potencia de 500W. Sus principales características son (Tabla 2):

Tensión de entrada	230V AC
Frecuencia de entrada	50 Hz
Eficiencia	70%
Temperatura de Funcionamiento	0 - 40 °C
Protecciones	OVP (Sobrevoltaje)
	OCP (Sobretensión)
	SCP (Cortocircuito)

Tabla 2. Características de la fuente

Como toda fuente de alimentación de estas características, ofrece las siguientes salidas (Tabla 3):

Color	Salida
	3.3 V
	5 V
	12 V
	-12 V
	-5 V
	GND

Tabla 3. Salidas de la fuente

7.2. Motores a pasos.

Un motor a pasos es un dispositivo electromecánico que convierte una serie de impulsos eléctricos en desplazamientos angulares discretos, lo que significa que es capaz de avanzar una serie de grados (pasos) dependiendo de sus entradas de control.

Los motores paso a paso son ideales para la construcción de mecanismos en donde se requieren movimientos muy precisos. Están compuestos por dos partes una fija, llamada estator y otra en movimiento, llamada rotor (Imagen 14). El estator se encuentra en la periferia del motor y es el encargado de generar el flujo principal, mientras que el rotor se encuentra en el centro del motor, unido al eje, y su función es reaccionar a la excitación del estator produciendo el movimiento de rotación o una fuerza de enclavamiento.

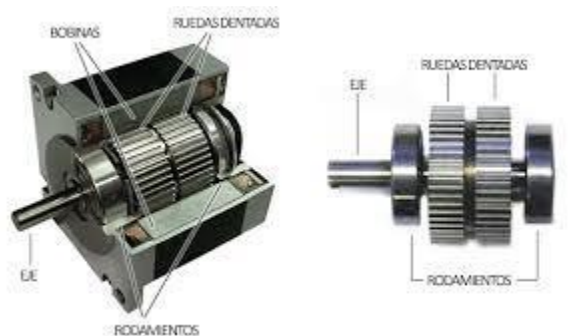


Imagen 14. Partes del motor a pasos

7.3. Drivers.

El A4988 es un controlador (drivers) que simplifica el manejo de motores paso a paso desde un autómata o procesador como Arduino. Estos controladores permiten manejar los altos voltajes e intensidades que requieren estos motores, limitar la corriente que circula por el motor, y proporcionan las protecciones para evitar que la tarjeta electrónica pueda resultar dañada.

Para su control únicamente requieren dos salidas digitales, una para indicar el sentido de giro y otra para comunicar que el motor avance un paso. Además, permiten realizar microstepping, una técnica para conseguir precisiones superiores al paso nominal del motor.

Dispone de protecciones contra sobreintensidad, cortocircuito, sobretensión y sobrettemperatura. En general, es robusto y fiable siempre que se realice la conexión correctamente, e incorporar disipadores de calor si es necesario. El A4988 es muy empleado en una gran variedad de proyectos que requieren el uso de motores paso a paso, como máquinas de CNC, plotters, robots que dibujan, impresoras 3D, y escáneres 3D. También son un componente frecuente en proyectos para controlar robots y vehículos, especialmente en aquellos que requieren variar de forma individual la velocidad de cada rueda.

El motivo de utilizar drivers es que los motores paso a paso de cierto tamaño y potencia, necesitan tensiones superiores a las que podrían soportar las bobinas por su corriente nominal. Por este motivo, los controladores incorporan un limitador de intensidad, que permiten alimentar el motor a tensiones nominales superiores a las que es posible por su resistencia e intensidad máxima admisible, en la imagen 15 se muestran las conexiones del Driver.

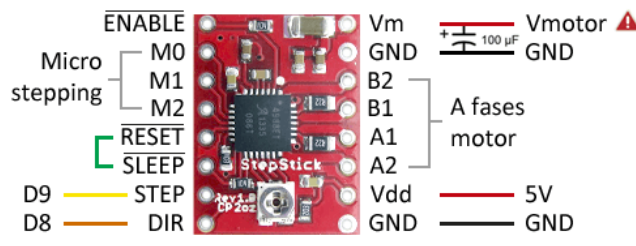


Imagen 15. Conexión del driver

Para no dañar el controlador y el componente, debemos seguir siempre el proceso rigurosamente sin saltarnos ningún paso.

- Conectar el controlador a tensión, sin el motor y sin microstepping.
- Medir con un voltímetro la tensión entre GND y el potenciómetro.
- Ajustar el potenciómetro hasta que la tensión sea el valor proporcionado por la fórmula.
- Apagar el montaje.
- Conectar el motor, interponiendo en medio un amperímetro.
- Realizar con cuidado el ajuste fino del potenciómetro, hasta que la intensidad sea la nominal del motor.
- Apagar el montaje.
- Retirar el amperímetro, y conectar el motor definitivamente.
- Conectar tarjeta electrónica al montaje.

7.4. CNC Shield.

La Shield es una pequeña placa que permite controlar hasta 4 motores paso a paso, soporta 4 controladores de potencia Driver A4988 o DRV8825, dispone de todas las conexiones necesarias para conectar interruptores de final de carrera, salidas de relé y diversos sensores.

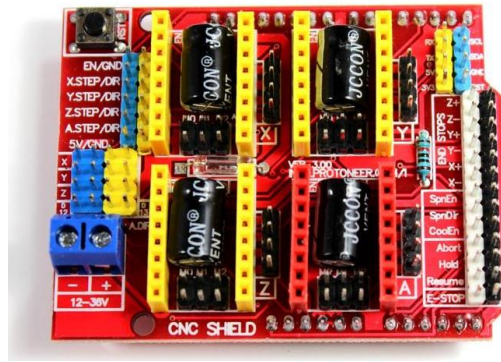


Imagen 16. CNC Shield

La CNC Shield cuenta con un diseño compacto, además cuenta con jumpers para control micro-stepping, en la tabla 3 podemos observar más características de la placa CNC Shield.

Alimentación	12-36 V
g.d.l. disponibles	4
Conexiones de finales de carrera	2 por eje
Salidas	“Spindle enable” “Coolant enable” “Direction”
Conexión de motores	Bornes tipo Molex de 4 pines
Drivers compatibles	A4988 y DRV8825
Dimensiones	6.9cm x 5.3cm x 1.9cm

Tabla 4. Características de la CNC Shield

8. Diseño del control.

8.1. Microcontrolador.

El FRDM-KL25Z es una plataforma de desarrollo de costo bajo para las MCU Kinetis L Serie KL2x (KL24 / 25) integradas en el procesador ARM® Cortex™ -M0+. Las características incluyen un fácil acceso a MCU I/O, funcionamiento con batería y baja potencia, un factor de forma basado en estándar con opciones de placa de expansión y una interfaz de depuración incorporada para programación flash y control de ejecución. El FRDM-KL25Z es compatible con una gama de software de desarrollo NXP y de terceros.

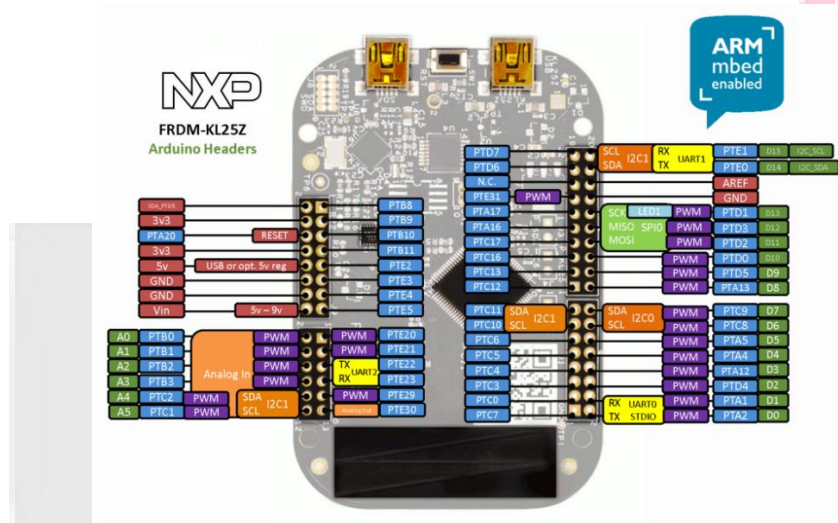


Imagen 17. Pines de Freescale KL25Z

8.2. Programador.

Mbed es una plataforma y un sistema operativo para dispositivos conectados a Internet basados en microcontroladores ARM Cortex-M de 32 bits. Dichos dispositivos también se conocen como dispositivos de Internet de las cosas. El proyecto es desarrollado en colaboración por Arm y sus socios tecnológicos.

8.2.1. Desarrollo de aplicaciones en Mbed.

Las aplicaciones para la plataforma Mbed se pueden desarrollar utilizando el IDE en línea de Mbed, un editor y compilador de código en línea gratuito. Solo se necesita instalar un navegador web en la PC local, ya que un proyecto se compila en la nube, es decir, en un servidor remoto, utilizando el compilador ARMCC C/C++. El IDE de Mbed proporciona espacios de trabajo privados con la capacidad de importar, exportar y compartir código con

el control de versión distribuido de Mercurial, y también se puede utilizar para la generación de documentación de código.

Las aplicaciones se pueden desarrollar también con otros entornos de desarrollo como Keil μ Vision, IAR Embedded Workbench y Eclipse con herramientas integradas GCC ARM.

8.2.2. Mbed OS

Mbed OS proporciona la plataforma y herramientas de software Mbed C / C ++ para crear firmware de microcontrolador que se ejecuta en dispositivos IoT. Consiste en las bibliotecas principales que proporcionan los controladores periféricos del microcontrolador, redes, RTOS y entorno de tiempo de ejecución, herramientas de compilación y scripts de prueba y depuración. Estas conexiones se pueden asegurar mediante bibliotecas SSL / TLS compatibles, como wolfSSL , que admite mbed-rtos.

Una base de datos de componentes proporciona bibliotecas de controladores para componentes y servicios que se pueden conectar a los microcontroladores para crear un producto final.

8.3. ROS.

Sistema Operativo Robótico (en inglés Robot Operating System, ROS) es un framework para el desarrollo de software para robots, que provee la funcionalidad de un sistema operativo en un clúster heterogéneo. ROS se desarrolló originalmente en 2007 bajo el nombre de *switchyard* por el Laboratorio de Inteligencia Artificial de Stanford para dar soporte al proyecto del Robot con Inteligencia Artificial de Stanford (STAIR).

Desde 2008, el desarrollo continua primordialmente en Willow Garage, un instituto de investigación robótico con más de veinte instituciones colaborando en un modelo de desarrollo federado.

ROS provee los servicios estándar de un sistema operativo tales como abstracción del hardware, control de dispositivos de bajo nivel, implementación de funcionalidad de uso común, paso de mensajes entre procesos y mantenimiento de paquetes. Está basado en una arquitectura en los nodos que pueden recibir, mandar y multiplexar mensajes de sensores, control, estados, planificaciones y actuadores, entre otros. La librería está orientada para un sistema UNIX(Ubuntu (Linux)). La instalación de ROS se muestra en el anexo 1.

8.4. Código de programación.

Primero se incluyen las librerías necesarias para poder hacer la conexión mediante nodos con el sistema operativo ROS.

```
#include "mbed.h"
#include <ros.h>
#include <std_msgs/Int16.h>
#include <std_msgs/Empty.h>
```

Posteriormente se hace la declaración de los nodos para poder establecer la comunicación con ROS y se declaran los pines de salida para hacer los pasos de los motores.

```
ros::NodeHandle nh;

DigitalOut step(D2);
DigitalOut dir (D5);
DigitalOut en (D8);
DigitalOut step1 (D3);
DigitalOut dir1 (D6);
DigitalOut step2 (D4);
DigitalOut dir2 (D7);
DigitalOut myled (LED3);
```

Se declaran las variables a utilizar para la recepción de datos tipo serial.

```
int paso=0, paso1=0, paso2=0, n=0;
int servo_1, servo_2, servo_3, juntos;
float stepDelay = 0.0004;
```

Se crean las funciones para la selección del nodo que se quiere mover.

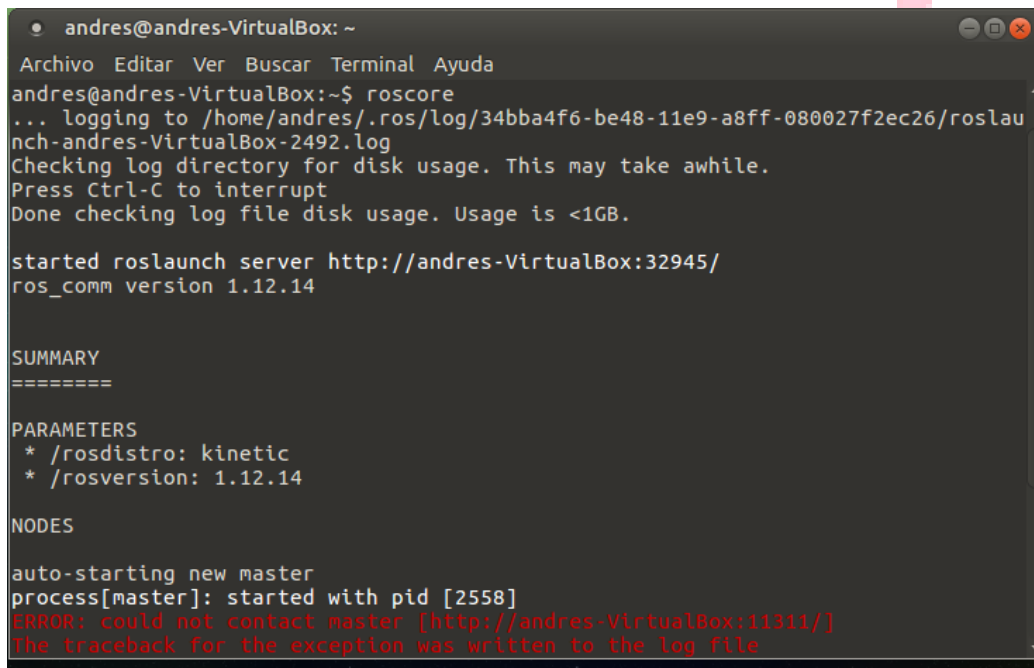
```
void motoresjuntos(const std_msgs::Int16& cmd_msg)
{
    juntos= cmd_msg.data;
    servo_1 = juntos
    servo_2 = (juntos/1.53);
    servo_3 = (juntos/17.05);
}
void servo_cb(const std_msgs::Int16& cmd_msg)
{
    servo_1 = cmd_msg.data;
}
void servo1_cb(const std_msgs::Int16& cmd_msg)
{
    servo_2 = cmd_msg.data;
}
void servo2_cb(const std_msgs::Int16& cmd_msg)
{
    servo_3 = cmd_msg.data;
}
void messageCb (const std_msgs::Int16& cmd_msg){
    myled = !myled;
}
```

Escribimos los nodos nuevamente para poder establecer la conexión con el microcontrolador.

```
ros::Subscriber<std_msgs::Empty> sub4 ("acción", &messageCb);
ros::Subscriber<std_msgs::Int16> sub2 ("motorz", servo2_cb);
ros::Subscriber<std_msgs::Int16> sub1 ("motorx", servo1_cb);
ros::Subscriber<std_msgs::Int16> sub ("motory", servo_cb);
ros::Subscriber<std_msgs::Int16> sub1 ("juntos", motoresjuntos);
```

8.4.1. Comandos desde ROS.

Primero inicializamos ROS desde la terminal.

A terminal window titled 'andres@andres-VirtualBox: ~' showing the output of the 'roscore' command. The output includes logging information, a check for disk usage, and the start of a roslaunch server. It also displays a summary of parameters (rostdistro: kinetic, rosversion: 1.12.14) and nodes (auto-starting new master, process[master]: started with pid [2558]). A red error message is visible at the bottom: 'ERROR: could not contact master [http://andres-VirtualBox:11311/]' and 'The traceback for the exception was written to the log file'.

```
andres@andres-VirtualBox: ~
Archivo Editar Ver Buscar Terminal Ayuda
andres@andres-VirtualBox:~$ roscore
... logging to /home/andres/.ros/log/34bba4f6-be48-11e9-a8ff-080027f2ec26/roslau
nch-andres-VirtualBox-2492.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://andres-VirtualBox:32945/
ros_comm version 1.12.14

SUMMARY
=====

PARAMETERS
* /rostdistro: kinetic
* /rosversion: 1.12.14

NODES

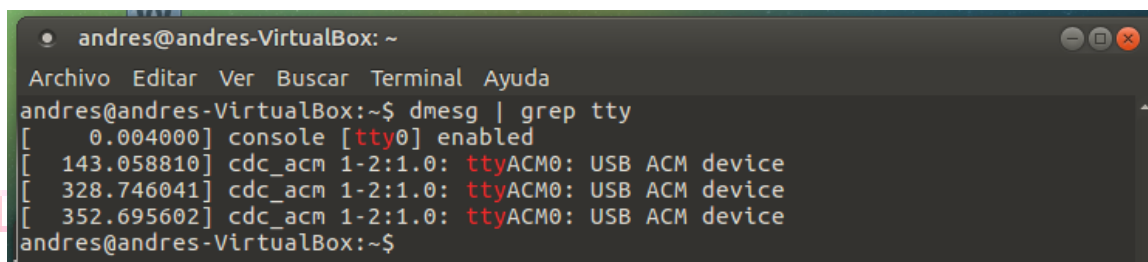
auto-starting new master
process[master]: started with pid [2558]
ERROR: could not contact master [http://andres-VirtualBox:11311/]
The traceback for the exception was written to the log file
```

Imagen 18. Roscore

Para saber el puerto serial que esta utilizando la tarjeta ejecutamos el siguiente código:

```
Dmesg | grep tty
```

La información desplegada (imagen 19) será la correspondiente al puerto que es ocupado por el microcontrolador.

A terminal window titled 'andres@andres-VirtualBox: ~' showing the output of the 'dmesg | grep tty' command. The output shows three lines of log messages indicating that the ttyACM0 device is enabled and that the USB ACM device is detected at the same location (cdc_acm 1-2:1.0).

```
andres@andres-VirtualBox: ~
Archivo Editar Ver Buscar Terminal Ayuda
andres@andres-VirtualBox:~$ dmesg | grep tty
[  0.004000] console [tty0] enabled
[ 143.058810] cdc_acm 1-2:1.0: ttyACM0: USB ACM device
[ 328.746041] cdc_acm 1-2:1.0: ttyACM0: USB ACM device
[ 352.695602] cdc_acm 1-2:1.0: ttyACM0: USB ACM device
andres@andres-VirtualBox:~$
```

Imagen 19. Puerto serial

Establecemos la conexión con el microcontrolador con el siguiente código:

```
roslaunch rosserial_python serial_node.py /dev/ttyACM0
```

Para activar los motores se utilizan los siguientes códigos. El cero indica la posición a la que queremos llegar, es decir, el 0 es sustituido por la posición deseada.

```
rostopic pub motory std_msgs/Int16 0 --once  
rostopic pub motorz std_msgs/Int16 0 --once  
rostopic pub motorx std_msgs/Int16 0 --once
```

9. Conclusiones.

9.1. Hernández Castillo Ana Yuritzi.

Este proyecto representa un trabajo en equipo a lo largo de un año, podría decir que es el primer proyecto en el que se ve la integración de mecatrónica como tal, fue un proyecto largo y complicado, pero al tener los resultados me sentí satisfecha de tanto.

Los robots manipuladores juegan un papel fundamental en las industrias, antes de presentarlo tuvimos la oportunidad de ir a grupo pisa para verlos en operación en el mundo laboral, fue muy satisfactorio verlo porque solo lo habíamos visto a escala y tal vez no teníamos una visión real de su funcionalidad a escalas reales.

Este proyecto creo una gran oportunidad de reforzar e incluso aprender cosas nuevas de lo antes aprendido .

9.2. Hernández García Andrés de Jesús.

Para empezar a ver la parte del control y la programación primero vimos cual microcontrolador utilizaríamos para esto, después de elegir el microcontrolador, empecé a investigar cual era la gramática para poder hacer la programación en la IDE del microcontrolador.

También hicimos una investigación detallada del programa ROS ya que necesitábamos conectar la tarjeta con dicho programa para realizar las acciones de los motores.

Se nos dificulto al principio ya que esto era algo nuevo que no habíamos implementado, pero después de investigar y realizar las tareas y prácticas que el profesor nos dejaba logramos nuestro objetivo.

9.3. Rodríguez Rodríguez José Luis.

La robótica es sin duda un área de interés de las investigaciones de la actualidad, con este proyecto se estudió sobre los robots manipuladores donde se pudo apreciar un desarrollo desde la identificación de una necesidad, hasta llevar a cabo la construcción de un prototipo, contemplando a grandes rasgos una solución con una estructura mecánica de brazo cartesiano de 3 ejes de libertad (CNC)

Durante el desarrollo del proyecto se realizaron modificaciones, en la estructura ya que la funcionalidad del mismo no era la adecuada y se necesitó rediseñar los ejes, además se

necesitó cambiar el controlador de Arduino a freescale k125z debido a lineamientos de la materia dinámica y control de robot,

Se cumplió con el objetivo del proyecto con éxito ya que el brazo cartesiano (CNC) funciona correctamente y es una herramienta de gran utilidad que se puede adaptar fácilmente a las actividades de una empresa manufacturera, partiendo de los comandos ingresados en el programa del mismo. Siendo esta una alternativa tecnológica al alcance de la pequeña y mediana empresa para automatizar el proceso de movimientos u conformado de piezas en prensas troqueladoras, incrementando la seguridad y productividad del proceso.

9.4. Rosales Ortiz Ian Alexis.

Este Proyecto tuvo como un objetivo final el realizar un equipo robótico capaz de tener movimiento en tres ejes cartesianos los cuales eran x, y, z, para llegar a ello se tuvo que realizar una serie de investigaciones.

Durante el procedimiento de este proyecto llegamos a utilizar programas de simuladores el cual nos mostrarían movimientos, esfuerzos y un análisis detallado del brazo, se pudo realizar esto con este gracias a buenas fuentes de información y un buen trabajo en equipo puesto que llegamos a tener programas que no hubiésemos visto o escuchado hablar los cuales eran Gazebo, Este lo podíamos encontrar en sistema operativo de Ubuntu para ello se realizó una partición o máquina virtual y otro programa era Blender igualmente este lo encontrábamos en el mismo sistema, esto fue lo que llegó a complicarnos un poco puesto que eran programas un poco complejos, al final encontramos la manera de manejar lo más básico de estos programas para llegar a nuestro objetivo deseado.

Para llegar al movimiento del brazo ya no virtualmente si no físicamente se utilizaron servo-motores, estos mediante la utilización de tarjeta Freescale y utilizando la Shield compuesta de drivers para controlarlos se llevó a cabo una programación un tanto compleja, pues se debía encontrar primeramente la manera de mover los servos y después de ellos darles movimientos a todos de una manera eficiente, se encontró que estos se pudieron mover conforme a pasos con un cierto límite para que estos no pegaran en la estructura.

Al final llegamos al objetivo a realizar con buena satisfacción hacia nosotros y posteriormente hacia el maestro, demostrando lo aprendido durante el cuatrimestre u otros anteriores y llevándonos con nosotros conocimientos y tal vez para nuevos estudiantes inspiraciones a realizar proyectos así.

10. Bibliografía.

- Alonso, J. A. (s.f.). *Robots Industriales*. Obtenido de http://platea.pntic.mec.es/vgonzale/cyr_0708/archivos/_15/Tema_5.4.htm
- Antonio, B., Peñín, L. F., Balaguer, C., & Aracil, R. (2019). *Fundamentos de Robótica*. Zapopan: McGraw-Hill.
- Caparroso, I., & Hernández, J. (1999). Una Introducción a la Robótica Industrial. *Facultad de Ingeniería*.
- Cazar, M. (2008). Diseño y Construcción de una Fresadora de 3 Ejes de Líneas Rectas. 150.
- Rentería, A., & Rivas, M. (s.f.). *Fundamentos de la Robótica Industrial*. McGraw-Hill.
- Reyes Cortés, F. (s.f.). *Robótica: Control de Robots Manipuladores*. Alfaomega.
- Rojas, J., Mahla, I., Muñoz, G., & Castro, D. (2003). Diseño de un Robot Cartesiano para Aplicaciones Industriales. *Revista Facultad de Ingeniería*, 116.

Anexos A. Instalando ROS en Ubuntu.

Descargue Ubuntu 16.04 LTS desde <http://releases.ubuntu.com/16.04> grabelo en un disco o escríbalo en una unidad USB usando *unetbootin* (<https://unetbootin.github.io>) para instalarlo en una PC. También puede instalar Ubuntu en VirtualBox (<https://www.virtualbox.org/wiki/Downloads>). Los tutoriales completos de instalación de Ubuntu están disponibles en www.ubuntu.com/download/desktop/install-ubuntu-desktop. Tenga en cuenta que en la configuración de VirtualBox, la memoria de video debe ser alta, y debe instalar los Complementos de VirtualBox Guest después de Instalación de Ubuntu.

A continuación, instale ROS en Ubuntu. Para obtener instrucciones detalladas sobre cómo instalar ROS Kinect en Ubuntu 16.04 desde el sitio web de ROS, vaya a <http://wiki.ros.org/kinetic/Installation/Ubuntu> . Si está interesado en ROS Indigo, utilice en su lugar <http://wiki.ros.org/indigo/Installation/Ubuntu> . También tenga en cuenta que, en lugar de Ubuntu 16.04, necesita 14.04 LTS para instalar ROS Indigo.

Una vez que se complete la instalación de ROS, ejecute el comando `roscore` para verificar que todo funciona bien. Una captura de pantalla de este comando se muestra en la imagen 25

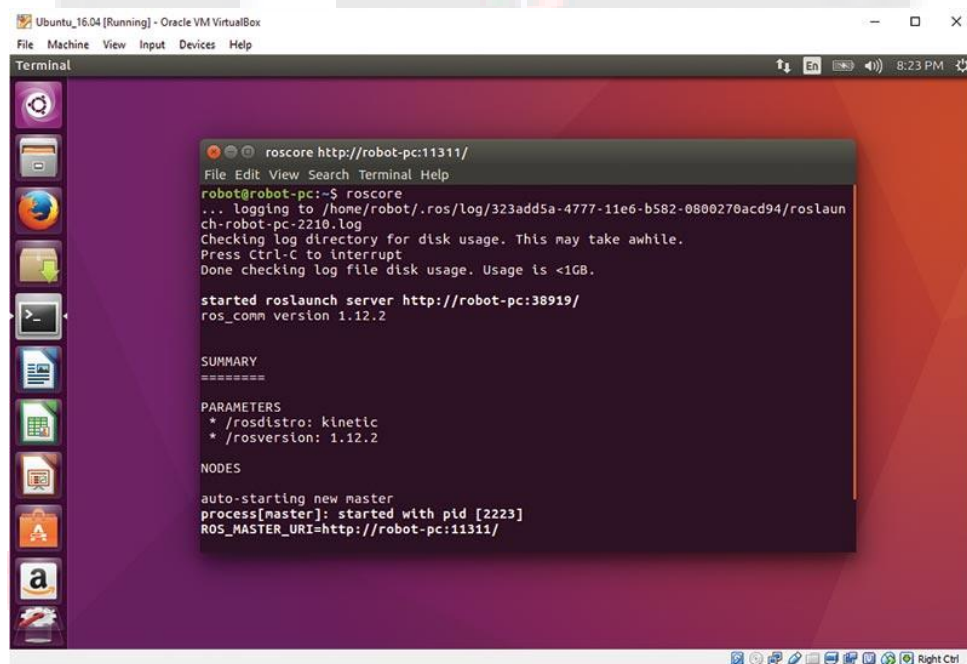


Imagen 20. Ejecutando roscore.

Instalación del paquete de interfaz Rosserial en Ubuntu

Después de configurar ROS en Ubuntu, tenemos que instalar los paquetes roserial en ROS. La instalación de un paquete se puede hacer de dos maneras:

Instalación a través de *apt-get* que instalará binarios pre-construidos.

Instalación mediante compilación de código fuente.

La forma más fácil de instalar paquetes es usar *apt-get*, pero en las últimas versiones de ROS, es posible que la mayoría de los paquetes no estén disponibles como binarios. En ese caso, podemos crear un espacio de trabajo ROS y descargar los paquetes de origen e instalarlo. Aquí está el procedimiento para construir un paquete roserial en ROS Kinect:

En su terminal, cree una carpeta vacía para el área de trabajo de ROS. Puedes darle cualquier nombre; Estoy usando **roserial_ws**:

```
mkdir -p ~/roserial_ws/src
```

Creating a folder called roserial_ws, and src folder inside the workspace folder<br

```
cd ~/roserial_ws/src
```

Switch to src folder<br

```
catkin_init_workspace
```

This will initialize a catkin workspace<br

```
git clone [url=https://github.com/ros-drivers/roserial ]https://github.com/ros-drivers/roserial [url];
```

Cloning latest source code of serial package in src folder<br

```
cd ~/roserial_ws
```

Change into workspace folder<br

```
catkin_make
```

Command to build the entire workspace

El comando *catkin_make* compilará todos los paquetes dentro del área de trabajo y generará carpetas adicionales como ' *build* ' y ' *devel* '. La carpeta de *compilación* contiene registros de compilación y la carpeta *devel* contiene scripts de shell y ejecutables generados. Para hacer que este paquete sea visible para el entorno de ROS, debe crear uno de los scripts de shell en el *devel* . El siguiente comando hará este trabajo:

```
echo "source ~/roserial_ws/devel/setup.bash" >> ~/.bashrc  
source ~/.bashrc
```

¡¡Felicidades!! Ya ha terminado con la instalación de origen de los paquetes roseriales !! Si está intentando instalar usando *apt-get* , puede usar este comando:

```
sudo apt-get install ros-kinetic-roserial
```

Nota: Si falla, cambie a la instalación de origen. Si está trabajando con ROS-Indigo, puede instalarlo directamente usando el siguiente comando:

```
sudo apt-get install ros-indigo-roserial
```