DISEÑO Y CONSTRUCCIÓN DE UN PROTOTIPO DE ROBOT DELTA CON IMPLEMENTACIÓN DE UN CORTADOR LÁSER CNC UTILIZANDO LA PLATAFORMA ROBOTIC OPERATING SYSTEM (ROS) PARA LA ELABORACIÓN DE ARTÍCULOS PUBLICITARIOS

Diana Tumbaco Mendoza, Wilmer Quimbita Zapata

Departamento de Energía y Mecánica de la Universidad de Las Fuerzas Armadas ESPE Extensión Latacunga

car_oli_s@hotmail.com
qwilmer09@gmail.com

Abstract— **this paper aims at construction, programming and simulation of a delta robot, laser print on soft materials, clip art or designed by the user in ROS**

The modeling was performed using inverse kinematics, obtaining the equations of the robot was used geometric method; these equations are used for the development of programming in Python. In programming, are used libraries like OpenCV for vectoring of images to draw. The display of the simulation is the rviz.

Finally, the development of this work shows the application of mechatronics with the ROS software, the software is free and the Ubuntu platform.

Resumen— **Este documento tiene como finalidad la construcción, programación, simulación de un robot delta, para imprimir con láser sobre materiales suaves, imágenes prediseñadas o diseñadas por el usuario en ROS.**

La modelación se realizó mediante la cinemática inversa, la obtención de las ecuaciones del robot se utilizó el método geométrico, estas ecuaciones son utilizadas para el desarrollo de la programación en Python. En la programación, se utiliza librerías como OpenCV para la vectorización de la o las imágenes a dibujar por el robot. El visualizador de la simulación es el Rviz propio del software ROS.

Finalmente, el desarrollo de este trabajo muestra la aplicación de la Mecatrónica con el software ROS, el software es libre y en la plataforma Ubuntu.

***Palabras Claves*— Robot Delta, ROS, OpenCV, Python, Ubuntu**

I. INTRODUCCIÓN

La Mecatrónica es una ingeniería que utiliza un sin número de áreas del conocimiento para el desarrollo de productos o sistemas, entre las primarias se pueden mencionar a la mecánica, la electrónica y la computación. Algunas de sus ramas secundarias son las matemáticas, la inteligencia artificial, la manufactura, la metrología, la robótica entre otras.

Los productos derivados de la aplicación de la Mecatrónica son óptimos y comercialmente competitivos y la mayoría de las aplicaciones se refiere a la generación de modelos y prototipos robóticos didácticos, podemos decir que un robot es un sistema mecatrónico por ello, el campo de la robótica es muy usado para la enseñanza de la misma.

El Sistema Operativo Robótico (Robotic Operating System, ROS) es un marco flexible para la escritura de software robot, con este software se puede controlar, simular los movimientos del robot en un ambiente virtual. Se trata de una colección de herramientas, bibliotecas y convenciones que tienen como objetivo simplificar la tarea de crear un comportamiento complejo y robusto robot en una amplia variedad de plataformas robóticas (Rey, 2013).

Dentro del campo de la robótica, los robots paralelos tipo delta, es un mecanismo compuesto de una plataforma móvil y una plataforma fija, conectadas por al menos dos cadenas cinemáticas, siendo una cadena cinemática la unión de dos o más eslabones (Kong and Gosselin, 2007). Generalmente, el número de grados de libertad del robot es igual al número de cadenas cinemáticas que lo conforman porque cada una de ellas es accionada por un actuador (Tsai, 1999) ((Urrea & Medina, 2012).Tienen ventajas sobre los robots antropomorfos, por ejemplo son más robustos y tienen gran precisión. Como desventajas los robots paralelos tienen un área de trabajo muy reducida y presentan grandes singularidades mecánicas. Los prototipos de robots paralelos son muy ilustrativos y motivantes en la enseñanza de la Mecatrónica

El desarrollo del prototipo, de un robot delta en ROS, es muy viable ya que hace parte de una plataforma experimental que tiene como objetivo facilitar y potenciar la enseñanza de: robots paralelos, control cinemático, sistemas avanzados de control y tele operación de robots utilizando el modelo y la simulación, aumentando la competitividad de los profesionales de la área de ingeniería Mecatrónica.

El robot delta dentro del Ecuador se realizado como proyecto de tesis el robot paralelo tipo STEWART en la ciudad de cuenca. ROBOTEK en una empresa ecuatoriana que distribuye robot “Pick and Place” esto a nivel de ecuador

en otros países se los hace como robot didáctico para el aprendizaje, industria de alimentos, empaquetado, etc.

Las aplicaciones importantes del robot paralelo son en el sector industrial: simuladores de vuelo y manejo de automóviles; en la medicina se aplica en cirugías, rehabilitaciones ortopédicas, etc. Como manipuladores se utiliza en posicionamiento de objetos y orientación. En nuestro caso es un robot dibujante por medio de láser, útil para la innovación de la robótica con este tipo de robot en nuestro país.

II. DESARROLLO

Descripción del robot

En la Figura 1.Se muestra el esquema de un robot tipo Delta, el cual consiste en tres brazos y dos plataformas: la fija, en la cual se ubican los actuadores y la plataforma móvil que porta el efector final del robot.

Figura 1. Esquema del Robot Delta
Elaborado por: Tumbaco, Diana y Quimbita, Wilmer

Los brazos se conectan por tres cadenas cinemáticas cerradas, cada brazo está conectado a un actuador separado 120º uno de otro, y está formado por dos eslabones. El eslabón inferior a su vez, está compuesto por un par de barras paralelas. Esta configuración restringe los movimientos del efector final a 3 translaciones de acuerdo con los ejes X, Y, Z.

Los motores están montados sobre la plataforma fija, y transfieren el movimiento a cada brazo mediante una articulación rotacional.

Cinemática Inversa

El objetivo es encontrar el ángulo de cada uno de los actuadores conociendo la posición del efector final. Por el diseño del robot, la articulación F_1J_1 ver figura 2 solo puede rotar en el plano YZ, formando una circunferencia con centro en el punto F_1 y con radio r_f. Por el contra, F_1, J_1 y E_1 son las llamadas articulaciones universales, lo que significa que E_1J_1puede rotar libremente en relación al punto E_1, formando una esfera con centro en E_1 y radio r_e.

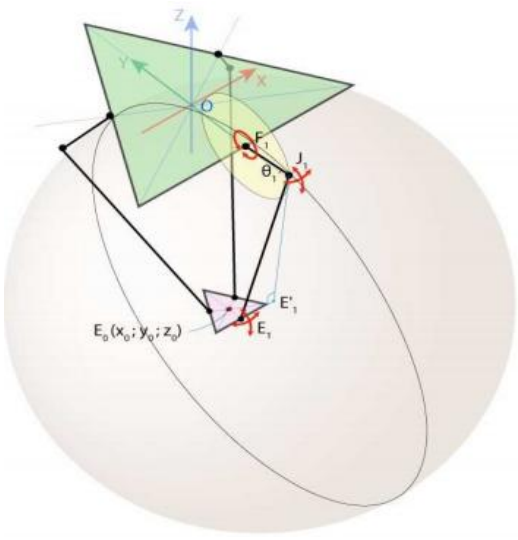

Figura 2. Articulación F1J1
Fuente: http://awesomebytes.com/wp-content/uploads/2012/09/Informe_delta_spfeiffer_agabas_con_anexos.pdf

La intersección de la circunferencia y la esfera se produce en dos puntos, se toma como solución el punto con menor valor en la coordenada Y. Al determinar la posición del punto J1 se puede obtener el ángulo θ1 del actuador, En la Figura 3 tenemos los parámetros geométricos para el cálculo de las coordenadas

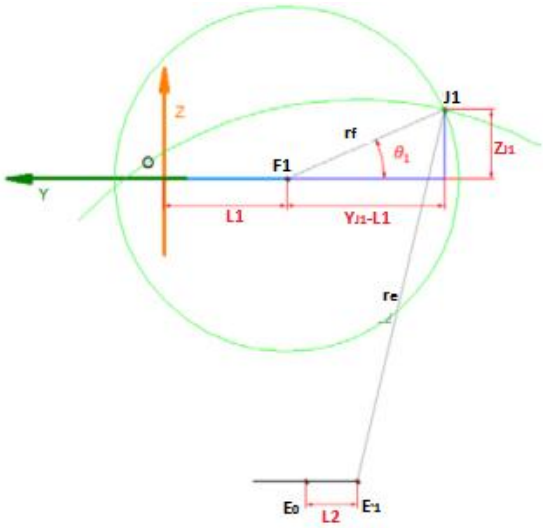

Figura 3. Vista lateral para análisis geométrico.
Fuente: http://awesomebytes.com/wp-content/uploads/2012/09/Informe_delta_spfeiffer_agabas_con_anexos.pdf

Coordenadas del Punto E_0,E_1, F_1yE'_1.

Primero realizamos una sustitución donde:

$$L_1 = y_{F1} = \frac{e}{2\sqrt{3}} \qquad L_2 = y_{E'1} = \frac{f}{2\sqrt{3}}$$

Con los datos anteriores tenemos las coordenadas de la siguiente manera:

$$E_0(x_0, y_0, z_0);\ E_1(x_0, y_0 - L_2, z_0)\ ;\ F_1(0, -L_1, 0);$$
$$E'_1(0, y_0 - L_2, z_0)$$

Con las coordenadas de los puntos descritos anteriormente, se plantea un sistema de dos ecuaciones no lineales que permiten encontrar la posición del punto J_1, con la cual se puede calcular el ángulo que forma el brazo con el plano horizontal, obteniendo así la solución esperada.

Sistema de ecuaciones:

$$(y_{J1} + \frac{f}{2\sqrt{3}})^2 + z_{J1}^2 = r_f^2 \qquad (1)$$

$$(y_{J1} - y_0 + \frac{e}{2\sqrt{3}})^2 + (z_{J1} - z_0)^2 = r_e^2 - x_0^2 \qquad (2)$$

Solucionando este sistema de ecuaciones (1) y (2) se llega a la siguiente ecuación cuadrática (3) que nos sirve para definir la solución.

$$a y_{J1}^2 + b y_{J1} + c = 0 \qquad (3)$$

Donde los valores de a, b y c son:

$$a = (1 + \frac{L_1 - y_0 + {L_2}^2}{z_0}) \qquad (4)$$

$$b = (2\left(\frac{L_1 - y_0 + L_2}{z_0}\right)\left(\frac{r_e^2 - x_0^2 - z_0^2 - r_f^2 - L_1^2}{2z_0}\right) - 2L_1) \qquad (5)$$

$$c = (\frac{r_e^2 - x_0^2 - z_0^2 - r_f^2 - L_1^2}{2z_0} - L_1^2 - r_f^2) \qquad (6)$$

Cuya solución general es de la forma mostrada en (7).

$$y_{J1} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \qquad (7)$$

Que tiene sentido solo cuando el argumento de la raíz cuadrada es positivo; de las dos posibles soluciones se toma la menor de las dos.
El valor del Angulo del brazo 1 se calcula con la formula (8)

$$\theta_1 = \arctan\left(\frac{z_{J1}}{y_{F1} - y_{J1}}\right) \qquad (8)$$

Para los otros brazos se usa la matriz de rotación con un ángulo de 120° para el brazo 2 y 240° para el 3. Esta matriz de rotación permite girar el sistema de coordenadas de manera que se pueda usar la solución descrita para el cálculo de los restantes ángulos.

PROGRAMACIÓN Y SIMULACIÓN EN ROS

Todos los elementos de programación ROS se desarrollaron en Python, para la cinemática inversa utilizamos las ecuaciones que definen al robot y que ya están desarrolladas, en cuanto a la simulación se utiliza la herramienta de visualización 3D Rviz de ROS, se ha optado por utilizar Markers, estos son formas simples (flechas, cubos, bolas, texto, etc.) ver figura 4 que representan los componentes del robot.

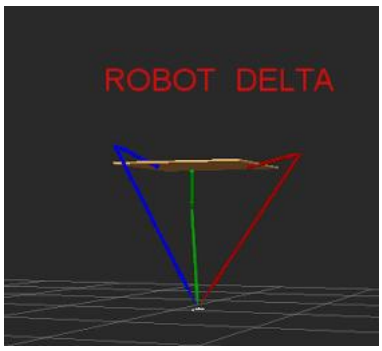

Figura 4. Salida gráfica del robot.
Elaborado por: Tumbaco, Diana y Quimbita, Wilmer

Para trabajar en ROS se construyó un espacio de trabajo donde se crea un paquete y se programa los scripts, el primer script contiene la cinemática del robot delta "cinematica.py", el siguiente script se llama "construccion_delta.py" archivo que consta de 3 funciones: crear_simulacion() la cual se encarga de crear el modelo del robot, colocar_pisicion_inicial() poner una posición inicial al modelo y para poder mover el modelo a las posiciones deseadas, está también la función mover_simulacion_al_punto (x, y, z, simulationMarkerArray) la cual podemos usarla para actualizar la posición de todo el modelo dejando el elemento terminal donde se cruzan los 3 brazos en el punto deseado.

En este mismo script también importamos la librería ya creada "cinematica.py", con esto es posible aplicar cinemática inversa para conocer la configuración total del robot de acuerdo a la posición a alcanzar.

Creación de la Aplicación.

Para esta aplicación utilizamos el paquete de interfaz de ROS con OpenCV, una biblioteca de funciones de programación de visión artificial en tiempo real, dentro del paquete también hay dos scripts llamado "drawing.py" donde está la opción de seleccionar, si se vectoriza o rasteriza la imagen todo esto con ayuda de OpenCV los puntos obtenidos

de la imagen se guardan en Ptos_Dibujo.txt. El otro script es "robot_delta_dibujador.py" se encarga de dibujar cada uno de los puntos en la posición deseada y donde se puede variar la escala "scale" de estos puntos que facilita la calidad de impresión de la imagen, además dentro de este archivo se puede variar el tamaño de la impresión.

El prototipo de robot delta debe dibujar una imagen cualquiera con formato JPG o PNG, y debe ser de calidad aceptable, pero necesariamente debe pasar por el procesado de imagen, para adaptar y reducir la complejidad de la misma. Para imágenes muy pequeñas, si bien el procesado es correcto, el dibujo no dará buenos resultados porque no estamos diseñando un robot para dibujar miniaturas.

Además dadas las restricciones de tamaño anteriores, el software que recibe la imagen para ser dibujada, recibe un archivo de imagen Figura 5.

Figura 5. Imágenes con formato PNG y JPG.
Fuente: http://start.iminent.com/StartWeb/3082/homepage/#q=homero%20simpson&s=images&p=1

Interfaz gráfica y proceso de vectorización y rasterización

La interfaz gráfica está construida para procesar automáticamente la imagen seleccionada cada vez que el usuario modifique un parámetro de procesamiento, ver Figura 6 y Figura 7. Esta visualización es la imagen que se guarda en el archivo Ptos_Dibujo.txt en forma de puntos y envía a la simulación del robot cuando inicia el proceso de dibujar.

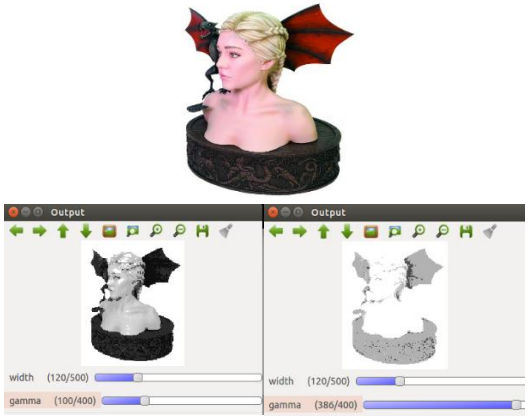

Figura 6. Imagen rasterizada.
Elaborado por: Tumbaco, Diana y Quimbita, Wilmer

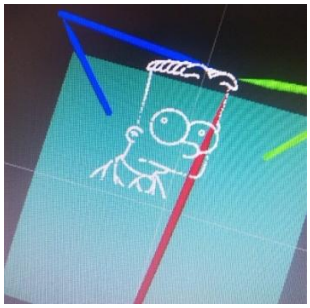

Figura 7. Imagen vectorizada en el Rviz.
Elaborado por: Tumbaco, Diana y Quimbita, Wilmer

Los parámetros width, gamma, smooth, apeture, lowThreshold, highThreshold inciden directamente en la velocidad del robot al momento de dibujar; por ejemplo, ajustando los parámetros de tal forma de obtener más calidad, se trazarán más segmentos y tomará más tiempo dibujarlos.

Implementación Electrónica, Eléctrica y Programación

El control de la posición se realiza automáticamente mediante el PID de los servos dynamixel.y para el control del láser se usa una tarjeta Arduino Uno, la cual también se integrará a ROS mediante su librería rosserial y apareciendo ros_lib en el IDE Arduino. Esta librería permite la comunicación con ROS. La figura 8 muestra la comunicación del sistema

Figura 8. Diagrama de conexiones del Robot Delta
Elaborado por: Tumbaco, Diana y Quimbita, Wilmer

Resultados

Se ha conseguido hacer dibujos en modo vectorizado en foamix figura 9, grabados en madera Figura 10.

Figura 9. Imagen Vectorizado.
Elaborado por: Tumbaco, Diana y Quimbita, Wilmer

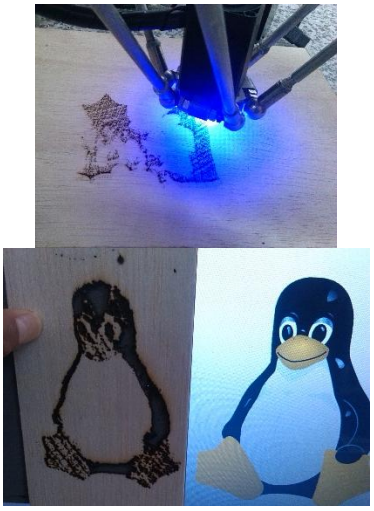

Figura 10. Imagen Rasterizada y dibujado con láser
Elaborado por: Tumbaco, Diana y Quimbita, Wilmer

III. CONCLUSIONES

- Para el diseño robot delta se utilizó software CAD solidworks y para el análisis de la estructura utilizamos Ansys.
- El material utilizado fue aluminio por sus características mecánicas en especial su bajo peso, resistencia a la deformación y facilidad de maquinabilidad.
- Robot Operating System facilita un desarrollo organizado de código reusable y abierto, con herramientas necesarias como Rviz y obtener una visualización del estado real del robot, así como los valores obtenidos por el análisis cinemático, de esta forma se puede hacer la comparación y corroborar que todo esté funcionando correctamente.
- Los Dynamixel al ser servos inteligentes y al interactuar directamente con ROS nos facilita su programación pero no se obtiene la precisión buscada, al momento de imprimir la imagen, la solución es utilizar la gama de los Dynamixel PRO con la diferencia que son de alto costo.
- La velocidad tanto en el corte como en el grabado disminuye cuando trabaja el simulador y el robot en tiempo real.
- El láser no corta colores claros porque se reflejan muy bien los colores claros lo contrario sucede con el color negro que absorbe mucha energía, por lo que quema rápidamente.
- El láser de 1 watt para grabar resultó una buena opción sobre todo para grabado en balsa y en aglomerado.
- La velocidad de grabado o corte depende del número de puntos enviados, entre más puntos sea el dibujo mayor será el tiempo.

IV. REFERENCIAS

[1] Essahli, Jamal Eddine ; Muñoz Ramos Jorge. (2012). Prototipo de robot paralelo "Delta-Robot". Recuperado el 27 de febrero del 2013, del Sitio Web del Universidad Politécnica de Valencia: http://riunet.upv.es/handle/10251/17925

[2] YO ROBOT (noviembre 26, 2006). Robótica Paralela Delta, recuperado de http://yorobot.wordpress.com/2006/11/26/robotica-paralela-delta/

[3] J. Breckling, Ed., *The Analysis of Directional Time Series: Applications to Wind Speed and Direction*, ser. Lecture Notes in Statistics. Berlin, Germany: Springer, 1989, vol. 61.

[4] Rey, P. (17 de 12 de 2013). *http://wiki.ros.org/*. Recuperado el 2014, de http://wiki.ros.org/

[5] Urrea, L., & Medina, S. (2012). *http://repository.unimilitar.edu.co/*. Obtenido de http://repository.unimilitar.edu.co/bitstream/10654/9953/1/UrreaMantillaLucasMateo2012.pdf

[6] Documentación ROS (Robot Operating System). Tomado de: ros.org. Recuperado el 27 de febrero del 2013 en http://www.ros.org/wiki/

[7] Mendez Canseco Mauricio C., Cado Robledo Carlos A., y Vargas Soto J. Emilio. “Desarrollo de un Robot Paralelo para Manufactura Ágil ” Septiembre 25 y 26, 2008, México D.F.

[8] Urbalejo Contreras Arturo, Jacobo Peña Javier, Jiménez López Eusebio, Quintero Esquer Jesús Alberto, López Cota Jorge Ricardo, Castro Bojórquez Julio Cesar. “Simulación, control y construcción de Robot Paralelo de 5 barras y 6 gdl para propósito didácticos”. 10º Congreso Nacional de Mecatrónica Noviembre 3 y 4, 2011. Puerto Vallarta, Jalisco.

[9] L. Ángel, R. Saltarén, N. Raguenes, J.M. Sebastián y R.Aracil. “Control visual de un robot paralelo: Análisis y diseño de la plataforma ‘robotenis’”. Madrid- España, recuperado el 25 mayo 2014.

[10] Alberto Traslosheros, José María Sebastián, Luís Ángel, Flavio Roberti, Ricardo Carelli. “Servo control visual para tareas de seguimiento dinámico tridimensional, mediante la utilización de una cámara en un robot Delta ”. recuperado: http://oa.upm.es/3651/1/INVE_MEM_2008_56343.pdf

[11] Pablo Barrera, Francisco Martín, Gregorio Robles, José M. Cañas and Vicente Matellán Héber Miguel Plácido Sobreira. "clever robot", Julio 2009.

[12] J.C. Reséndiz B., A. J. Reséndiz B., J. Pérez M, y J.G. Suárez-Romero. “Aplicación y desarrollo trigonométrico de las ecuaciones del movimiento de un robot paralelo plano de tres grados de libertad”, 8vo Congreso Iberoamericano de Ingeniería Mecánica. Cusco, 23 al 25 de Octubre de 2007

[13] Lucas mateo Urrea Mantilla, Sergio Alejandro medina papagayo. “Diseño e Implementación de una plataforma robótica tipo Delta” , Universidad Militar Nueva Granada, Facultad de ingeniería, Programa de ingeniería en mecatrónica, Bogotá, d.c. 2012.

[14] Brian Gerkey Steve Cousins e Willow Garage. "sharing software with ros". IEEE JOURNALS, junio 2010.

[15] Steve Cousins. "Exponential growth of ros". IEEE JOURNALS, Marco 2011.

[16] Robotic Operating System. Adaptado de http: //www.ros.org/wiki/, accedido última vez 23/05/2014.

[17] Aplicaciones a través de computador para robot delta (en ROS). Recuperado: http://awesomebytes.com/2012/09/13/computer-aplications-for-a-delta-robot-via-ros/

[18] Rubiano J F, Peña C A, Martínez E; “Avances en el desarrollo de una plataforma de control mental de un robot paralelo tipo delta ”, VII Congreso Bolivariano de Ingeniería Mecánica, Cusco, 23 al 25 de Octubre del 2012