Laboratorio 3 - Cinemática Inversa - Robot Phantom X - ROS

1. Objetivos

- Determinar el modelo cinemático inverso del robot *Phantom X*.
- Generar trayectorias simples a partir del modelo cinemático inverso del robot.
- Implementar el modelo cinemático inverso del robot en MATLAB.
- Operar el robot usando ROS a partir de trayectorias generadas en MATLAB o Python.

2. Requisitos

- Ubuntu versión 20.xx preferible 20.04 LTS con ROS.
- Espacio de trabajo para *catkin* correctamente configurado.
- Paquetes de Dynamixel Workbench. `https://github.com/fegonzalez7/rob_unal_clase3`
- Paquete del robot Phantom X: `https://github.com/felipeg17/px_robot`.
- MATLAB 2015b o superior instalado en el equipo.
- Robotics toolbox de Mathworks (Disponible desde la versión 2015 en adelante).
- Toolbox de robótica de *Peter Corke*.
- Piezas tipo 1 y 2 del proyecto (ver planos adjuntos).

Figura 1: Robot Phantom X Pincher.

3. Descripción

3.1. Cinemática Inversa

El problema cinemático inverso consiste en determinar la configuración articular de un manipulador, dadas la posición y orientación del efector final respecto a la base. Este problema puede resolverse mediante métodos geométricos,

algebraicos o numéricos. En el caso particular del robot *Phantom X* el cual posee 4 GDL, el enfoque más práctico es combinar el método geométrico con el desacople de muñeca.

4. Ejercicio en el laboratorio

MATLAB + Toolbox:

- Siguiendo la metodología analizada en clase desarrolle un modelo de cinemática inversa del manipulador *Phantom X* e impleméntelo en MATLAB. Se recomienda usar el *toolbox* para verificar la solución hallada.
- Esboce el espacio de trabajo del robot *Phantom X*.
- Consulte los métodos disponibles en el *toolbox* para determinar la cinemática inversa de un manipulador.

Análisis:

- Sabiendo que el robot *Phantom X* posee 4 GDL, de los cuales 3 corresponden a posición, el GDL restante proporciona una medida independiente para un ángulo de orientación (asuma orientación en ángulos fijos). ¿De qué ángulo de orientación se trata?
- ¿Cuántas soluciones posibles existen para la cinemática inversa del manipulador *Phantom X*?
- Consulte en qué consiste el espacio diestro de un manipulador.

ROS - Aplicación de Pick and place:

- Se tiene como objetivo implementar una aplicación de Pick And Place con el robot *Phantom X*, que consiste en lo siguiente:
  1. El robot deberá tomar la pieza tipo 1 que se encuentre a su derecha y ubicarla al frente.
  2. El robot deberá tomar la pieza tipo 2 que se encuentra a su izquierda, e insertarla en la pieza 1 que está ahora al frente

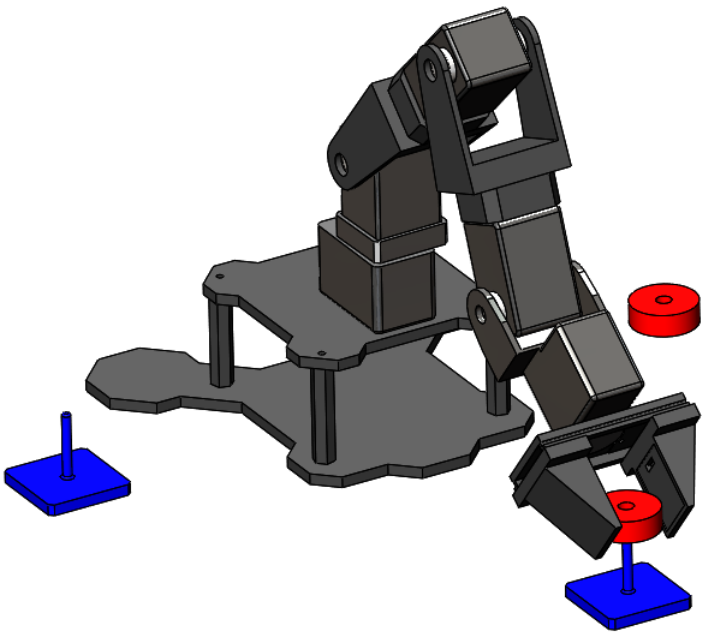

Figura 2: Proceso de ubicación de piezas usando robot *Phantom X*.

- Tomando como base el trabajo realizado en el laboratorio de cinemática directa, y utilizando el modelo de cinemática inversa desarrollado, cree un *script* que ejecute la rutina descrita. Puede realizarlo por medio de MATLAB o Python, según sea su preferencia.
- **Restricciones:**
  - Las trayectorias deberán ser tipo *pick and place*, esto es, recorridos en forma rectangular, movimientos verticales para subir y bajar, y movimiento horizontal para realizar desplazamientos.

ROS - Aplicación de movimiento en el espacio de la tarea

- De manera similar al trabajo realizado en el laboratorio de cinemática directa, cree un *script* (en Matlab o Python, según se prefiera) que permita mover el robot de manera escalada, con la salvedad de que no de manera articular como anteriormente se hizo, sino de forma operacional, es decir, en el espacio de la tarea del efector final del robot (en x, en y, en z, etc). El *script* debería seguir la siguiente lógica:
  - Se debe realizar el movimiento por pasos, es decir, se debe establecer un avance (en traslación o en orientación, según sea el caso) para que el robot realice ese avance en el eje que se solicitó.
  - Se le debe indicar al programa qué tipo de movimiento se desea realizar. Se debe imprimir en consola el nombre del movimiento operativo. El tipo de movimiento se cambia de la siguiente manera:
    - Los tipos de movimiento son los siguientes, junto con el nombre que se debe imprimir en pantalla para cada uno:
      1. Traslación en X - *trax*
      2. Traslación en Y - *tray*
      3. Traslación en Z - *traz*
      4. Rotación sobre el eje O del TCP - *rot*
    - Con la tecla 'W' se pasa al siguiente tipo de movimiento. Por ejemplo, si está en *trax*, pasaría a *tray*, y así.
    - Con la tecla 'S' se pasa al tipo de movimiento anterior. Por ejemplo, si está en *rot*, pasaría a *traz*, y así.
    - Se puede realizar de manera cíclica, es decir, que el siguiente tipo de movimiento a *rot* sea *trax*, y que el anterior de *trax* sea *rot*.
  - Al pulsar la tecla 'D', el efector final debería dar un salto igual al avance establecido por el equipo, en la dirección del tipo de movimiento seleccionado y en sentido positivo.
  - Al pulsar la tecla 'A', el efector final debería dar un salto igual al avance establecido por el equipo, en la dirección del tipo de movimiento seleccionado y en sentido negativo.
  - Obtenga la visualización del manipulador en RViz, de tal manera que se evidencien todos los movimientos realizados en el espacio de la tarea. **Nota:** Los movimientos en el espacio de la tarea **siempre** se realizan con respecto al sistema de referencia de la base.

Observaciones

1. **Entrega: 20/05/2022**
2. **Forma de trabajo:** Grupos de laboratorio
3. Se debe crear un repositorio en Github en donde se expliquen los materiales y métodos, los resultados obtenidos, los análisis realizados y las conclusiones. Es importante incluir estos aspectos ya que constituyen una parte importante en el proceso de aprendizaje.
4. Los puntos que requieran implementación de funciones deberán tener comentarios de cómo se utilizan y adjuntar archivos **.m** y **.py** al repositorio. También se deberán adjuntar los demás archivos relevantes que hayan sido creados y utilizados, como **.launch, .yaml, .urdf, .rviz**, entre otros.

5. Se deberá incluir vídeos en los que se muestre el funcionamiento de los *scripts* generados en Matlab o Python para cada una de las aplicaciones propuestas con ROS. Los vídeos deberán ser subidos a *Youtube* y poseer los respectivos enlaces en el repositorio. Este ítem es de **CARÁCTER OBLIGATORIO** y equivale al 20 % de la nota del laboratorio.

Referencias

[1] Mark W. Spong, Seth Hutchinson, and M. Vidyasagar. *Robot Dynamics and Control*. 2004.

[2] Peter Corke. *Robotics Toolbox for Matlab, Release 9*. 2015.

[3] Martinez, A., Fernandez E. *Leaning ROS for robotics programming, PackT Publishing*. 2015.