

Practica 3

Cinemática de Robots



UPZMG

Avalos Lupercio Jesús Jail

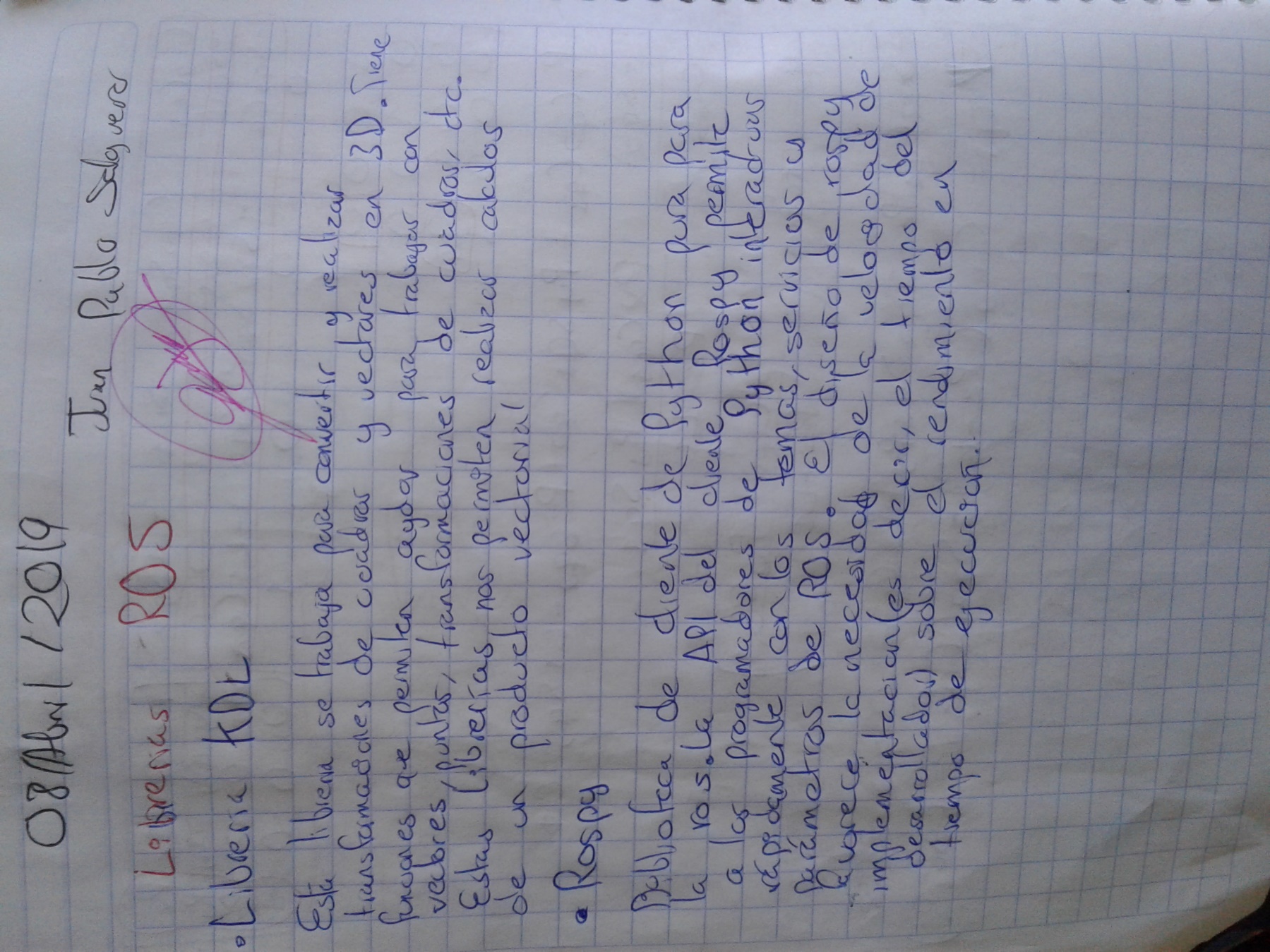
García Barajas Raul Israel

Martínez Jacinto Ricardo

Rubio García Rodrigo

Salguero Hernández Juan Pablo

8º A

**

*Librería de ROS utilizada en el robot*

***Librería KDL***

Esta librería se trabaja para convertir y realizar transformaciones de cuadros y vectores en 3D. Tiene funciones que permiten ayudar para trabajar con vectores, puntos, transformaciones de cuadros, etc.

Estas librerías nos permiten realizar los cálculos de un producto vectorial, también para transformar un punto en un marco de referencia diferente o incluso cambiar el punto de referencia diferente o incluso cambiar el punto de referencia de un giro.

La cinemática y dinámica representan gran importancia ya que pueden representar una cadena cinemática mediante un objeto y usar los solucionadores KDL para cualquier cosa que se necesite.

Y en esta ocasión nuestro robot presenta estos posibles inconvenientes, llamados así por el hecho de que se tardaba un tiempo considerable en obtener las MTH y ya con esta librería, todo será más rápido de obtener, Al ser un robot antropomórfico, se presenta la opción de tener el objeto en 3D utilizando todos los espacios posibles en las coordenadas, así que con esta librería activa se nos facilitaría el poder transformarlos, los cálculos requeridos tales como encontrar las matrices de los movimientos y medidas es cosa que también de manera automática o cuando se requieren podemos obtenerlas con KDL y es así como de manera más sencilla tenemos hasta la propia ayuda de la ROS cuando se presenten algunos inconvenientes errores en la cinemática que estaba y o la dinámica del programa que se quiere comenzar.

**Rospy**

rospy es una biblioteca cliente de Python pura para ROS. La API del cliente Rospy permite a los programadores de Python interactuar rápidamente con los temas, servicios y parámetros de ROS. El diseño de rospy favorece la velocidad de implementación (es decir, el tiempo del desarrollador) sobre el rendimiento en tiempo de ejecución, de modo que los algoritmos se puedan prototipar y probar rápidamente dentro de ROS. También es ideal para el código de ruta no crítica, como el código de configuración e inicialización. Muchas de las herramientas de ROS están escritas en rospy para aprovechar las capacidades de introspección de tipo. Muchas de las herramientas de ROS, como rostopic y rosservice , están construidas sobre rospy.

Esta biblioteca permite la comunicación de los nodos en Python locual es útil cuando se programa en la raspberry pi 3.

|  |  |  |
| --- | --- | --- |
| calibración\_estimación | Vincent Rabaud, Michael Ferguson | Ejecuta una optimización para estimar los parámetros cinemáticos de un robot. |
| cl\_transforms | Gayane Kazhoyan, Georg Bartels | Biblioteca de transformación homogénea para Common Lisp. |
| conversiones propias | Tully Foote, Adam Leeper | Funciones de conversión entre: - Eigen y KDL - Eigen y geometry\_msgs. |
| turtlesim | Dirk Thomas | turtlesim es una herramienta hecha para enseñar paquetes de ROS y ROS. |
| pid | Andy Zelenak, Paul Bouchier | Ejecutar un nodo de control PID |
| rosserial\_python | Paul Bouchier, Mike Purvis | Una implementación basada en Python del protocolo rosserial. |
| servidor\_sererial | Mike Purvis | Una alternativa de servidor más orientada al rendimiento y la estabilidad implementada en C ++ para rosserial\_python. |
| control de rotores | Fadri Furrer, Mina Kamel, Mina Kamel, Janosch Nikolic, Markus Achtelik | Paquete de control de rotor |
| single\_joint\_position\_action | ROS huérfanos mantenedores de paquetes | la acción de posición de una sola articulación es un nodo que proporciona una interfaz de acción para ordenar una trayectoria para mover una articulación a una posición particular. La acción informa de éxito cuando la articulación alcanza la posición deseada. |

**Comandos útiles en ROS**

Utilizando rospack

rospack le permite obtener información sobre los paquetes. En este tutorial, solo cubriremos la opción de búsqueda, que devuelve la ruta al paquete.

Uso:

$ rospack find [package\_name]

Ejemplo:

$ rospack find roscpp

volvería:

YOUR\_INSTALL\_PATH / share / roscpp

Si instalara ROS Kinetic desde apt en Ubuntu Linux, vería exactamente:

/ opt / ros / kinetic / share / roscpp

**Roscore**

Roscore es una colección de nodos y programas que son requisitos previos de un sistema basado en ROS. Usted debe tener un roscore en marcha con el fin de ROS nodos se comuniquen. Se inicia utilizando el comando roscore.

NOTA: Si usa roslaunch, iniciará automáticamente Roscore si detecta que aún no se está ejecutando.

Uso

El roscore se puede iniciar usando el ejecutable roscore:

Roscore

**Rosservice**

contiene la herramienta de línea de comandos rosservice para enumerar y consultar los servicios de ROS. También contiene una biblioteca de Python para recuperar información sobre los Servicios e invocarlos dinámicamente. La biblioteca de Python es experimental y es solo para uso interno.

**Rosnode**

es una herramienta de línea de comandos para mostrar información de depuración sobre los Nodos ROS, incluidas publicaciones, suscripciones y conexiones. También contiene una biblioteca experimental para recuperar información de nodo. Esta biblioteca está destinada únicamente para uso interno.