1. Introducción

En este proyecto de fin de carrera se ha realizado un sistema de monitorización en tiempo real del movimiento de un brazo humano. Además, para comprobar el funcionamiento del sistema, se han creado tres aplicaciones que hacen uso de dicho sistema de monitorización.

1.1. ¿Qué es un sistema de monitorización?

Entiéndase por sistema de monitorización el conjunto de elementos tanto de hardware (sensores) como de software (programas para captura de datos, procesamiento, visualización, etc) que nos permiten conocer una serie de magnitudes de cierto entorno. En este caso, el objetivo del sistema de monitorización es averiguar la posición en tiempo real de un brazo humano.

1.2. ¿Qué es la posición de un brazo humano?

Se idealizará el brazo humano como un sistema articulado formado por tres articulaciones (hombro, codo y muñeca), y tres eslabones (brazo, antebrazo y mano). En este proyecto se definirá la posición de un brazo como el conjunto de magnitudes que permiten determinar la posición y orientación de cada eslabón y el estado de rotación de cada articulación, y a partir de las cuales se podrán obtener otras magnitudes derivadas de estas (ángulos, velocidades, aceleraciones, etc.).

1.3. ¿Qué aplicaciones se crearán?

Además del sistema de monitorización en sí, se han creado tres ejemplos de aplicaciones en los que se hace uso del sistema de monitorización.

1. **Visualización de un modelo simplificado del brazo en un simulador 3D**: En este programa se podrá visualizar la posición del brazo en tiempo real.

2. **Simulacíon de un modelo de un brazo robótico**: Haciendo uso del sistema de monitorización se podrá controlar la posición de una simulación de un brazo robótico.

3. **Teleoperación de un robot real**: también se utilizará el sistema de monitorización para mover el brazo robótico de un robot real mediante el movimiento del brazo humano.

Adicionalmente se han creado una serie de librerías y utilidades que facilitan en gran manera la toma de datos de los sensores, la determinación de posiciones, orientaciones, ángulos y más magnitudes a partir de los datos capturados, y la visualización y simulación en otros programas, y que pueden ser utilizados en futuras aplicaciones.

Por último cabe mencionar que una de las características que se persiguieron en este proyecto es la modularidad. Cada parte del sistema es posible sustituirla por otra diferente sin apenas cambios en las demás.

2. Primer objetivo: Captura de datos

El primer problema que se trató de afrontar es la creación de un sistema de captura de datos de unos sensores para su posterior procesamiento en un PC.

2.1. Selección de sensores

Para cumplir el objetivo del proyecto buscamos un tipo de sensores que no sólo permitan determinar la posición del brazo, si no que además sean de fácil colocación en el brazo, que sean ligeros y que no dificulten el movimiento en absoluto de la persona.

Entre los muchos tipos de sensores existentes capaces de determinar la posición de un sistema articulado, destaca una clase que cumple todos las características

deseadas, que son las unidades de medición inercial o IMUs. Los IMUs son dispositivos electrónicos que hacen uso de un conjunto de acelerómetros, giróscopos y magnetómetros para determinar velocidad, orientación y otros parámetros.

En el laboratorio de robótica se dispone de una red de IMUs de la marca Xsens MTx, que pueden conectarse al PC mediante conexión USB formando una red mediante un máster Xbus. Las ventajas que se obtienen con el uso de estos sensores son las siguientes:

- Posibilidad de incorporar un número variable de sensores al sistema de monitorización de forma sencilla. Esta característica permitirá extender la monitorización a sistemas articulados a otros sistemas con más de 3 eslabones.

- Obtención directa de la orientación del cuerpo al que están acoplados. Los sensores MTx incorporan un filtro de Kalman extendido para la obtención de la orientación. Con esto se logra un error mínimo en la determinación de la orientación ($< 1^\circ$ para sistemas estáticos y $< 2^\circ$ para sistemas dinámicos). Además incorporan una corrección de la orientación utilizando los magnetómetros y acelerómetros de forma que se eliminan los problemas de deriva característicos de las unidades inerciales.

2.2. Captura de datos en un PC

Una vez escogidos los sensores que se utilizarán, nos encontramos con el problema de capturar los datos que proporcionan los sensores para poder utilizarlos en un ordenador. Para la obtención de los datos del puerto serie será necesario utilizar un programa informático llamado driver, en el que se implemente el protocolo de comunicaciones de bajo nivel que hace uso el dispositivo de hardware conectado al ordenador.

En la documentación de los sensores Xsens hay disponibles una serie de archivos de código fuente en C++ en las que se implementan una serie de clases, estructuras

y funciones que resuelven la comunicación a bajo nivel con los sensores Xsens y el máster Xbus, por lo que por suerte no se ha tenido que realizar el driver partiendo de cero. Sin embargo, el código es muy extenso y complejo, y su uso no es muy intuitivo, por lo que se ha decidido encapsular toda la complejidad de las clases de bajo nivel en una clase que permita realizar operaciones con los drivers, como su configuración y lectura de datos mediante llamadas a funciones de forma sencilla.

3. Segundo objetivo: Comunicación entre programas

En la anterior etapa del proyecto se había conseguido resolver la captura de datos de la red de sensores Xsens MTx mediante un programa driver creado sobre el código disponible en la documentación. Dada la modularidad de la que se pretende dotar al sistema, es preciso un sistema que haga posible que distintos programas se encarguen de distintos problemas y además exista comunicación entre ellos. Esto es, que por ejemplo distintos programas puedan obtener los datos del driver de los sensores.

3.1. La plataforma ROS

Cinco años atrás nace la plataforma ROS (Robot Operating system), con el objetivo de ayudar al desarrollador en la creación de software para robots. Actualmente es la plataforma líder de este tipo, y proporciona una cantidad enorme de herramientas como librerías para distintos lenguajes, drivers, programas de visualización de datos, simulación 3D, comunicaciones, servidor de parámetros, entre muchas otras. ROS es distribuido bajo una licencia BSD como software libre y gratuito. Su filosofía principal es proporcionar a los desarrolladores una plataforma común para colaborar y compartir proyectos. El resultado de esto es que aparte de las herramientas proporcionadas por ROS, actualmente existen miles de paquetes, creados

por los propios usuarios y empresas, con software con librerías, drivers para distintos componentes de hardware como sensores y robots, y multitud de otras herramientas disponibles para funcionar en ROS y de instalación muy sencilla. Veremos cómo el uso de esta plataforma nos proporcionará solución a muchos problemas a lo largo de este proyecto.

3.2. ¿Cómo ayuda ROS en la creación de programas?

En ROS todo el software está organizado en paquetes. Cada paquete puede contener distintos programas (llamados nodos) y librerías, normalmente con una función en común. El uso de paquetes proporciona una manera muy sencilla de compartir y reutilizar los distintos programas. ROS proporciona herramientas para la creación de paquetes de software de forma automática. Además sus herramientas están implementadas de forma nativa en varios lenguajes de programación (C++, python, lisp, octave, y con más pensados para implementar a corto plazo), de forma que la creación de programas que funcionan en ROS es muy natural.

3.3. ¿Cómo ayuda ROS en la comunicación entre programas?

En ROS, los distintos nodos en ejecución pueden comunicarse entre si mediante el envío de mensajes. Los mensajes es una estructura de datos que pueden contener valores numéricos, arrays, strings, etc. El paso de mensajes se hace a través de los llamados topics, que son canales de comunicación. Cada topic acepta un tipo de mensaje y tiene un nombre que sirve para identificarlo. Para realizar la comunicación entre dos nodos, uno de ellos publicará un mensaje en cierto topic, mientras que otro nodo estará suscrito a dicho topic y recibirá el mensaje cada vez que el primero lo publique. De esta forma se puede crear una red de nodos en la que cada uno de ellos se dedique a cierto proceso. El proceso principal de ROS, llamado máster,

se encargará él mismo de gestionar las comunicaciones, llevando un registro de los distintos nodos y topics.

3.4. ¿Qué más nos proporciona ROS?

ROS además ofrece la posibilidad de distribuir los nodos en distintas unidades de procesamiento que estén conectadas a la misma red de forma extremadamente sencilla (sólo será preciso modificar una variable de entorno a la hora de ejecutar cada nodo). De esta forma podemos aprovechar la disponibilidad de unidades con mayor capacidad de procesamiento de datos para ejecutar nodos más exigentes, y dejar en las unidades con menor capacidad procesamiento (como robots) los drivers necesarios. Aprovecharemos esta capacidad de ROS en el ejemplo de la teleoperación del robot.

Por último cabe mencionar algunas herramientas de ROS que haremos uso en las aplicaciones, como el programa Gazebo, que es un simulador de sólidos rígidos que proporciona un visualizador 3D de gran potencia, o la herramienta rxplot, que permite realizar una representación gráfica de los datos de los topics en tiempo real, o los archivos bag, que proporcionan una manera de grabar datos de ciertos topics para reproducirlos en otro momento o lugar de forma fiel.

3.5. Adaptación del driver Xsens para trabajar en ROS

Para completar el driver de los sensores Xsens, se ha creado un nodo de ROS en el que se ha incluido la clase que se ha mencionado anteriormente. Este nodo utiliza dicha clase para realizar la configuración de los sensores conectados, y publicará los datos leídos en cada ciclo en distintos topics de ROS.

4. Tercer objetivo: Creación de las aplicaciones

Una vez creado un sistema para capturar la posición del brazo en tiempo real que puede capaz de comunicarse con otros programas, la siguiente fase del proyecto es crear las aplicaciones que hacen uso de este sistema. Cada aplicación consistirá en varios nodos comunicándose con el nodo del driver de los sensores, con un objetivo específico. Para extender con la modularidad a las aplicaciones, se creará una arquitectura en la que se distinguirán dos clases de nodos.

- **Nodos de procesamiento de datos**: Son los que recogerán los datos de los sensores y los transformarán en otras magnitudes más útiles para la aplicación.
- **Nodos actuadores**: Los constituyen los drivers de los robots que se pretendan mover, o los programas de visualización como Gazebo, en los que se modificará la posición de un modelo.

4.1. Primera aplicación: Modelo simplificado de un brazo en el programa Gazebo

Esta primera aplicación consiste en utilizar el programa Gazebo para realizar una visualización en tiempo real de la posición del brazo humano. Para capturar la posición del brazo se utilizan tres sensores, que obtendrán las orientaciones del brazo, antebrazo y mano. En este ejemplo, el nodo procesador será el encargado de calcular las posiciones espaciales de cada eslabón del brazo mediante los cuaterniones de orientación leídos de los topics publicados por el nodo sensor, que realizará la configuración de los sensores para que den por salida cuaterniones de orientación. El nodo procesador además creará los mensajes que le pasará al nodo actuador para indicarle el estado del brazo. El nodo actuador es en este caso el visualizador Gazebo, que con el mensaje recibido del nodo procesador cambiará la posición del modelo del brazo.

4.2. Segunda aplicación: Simulación del brazo robótico del robot YouBot

En este segundo ejemplo se muestra como realizar el control de una simulación de un brazo robótico de un robot utilizando de nuevo los sensores Xsens. El modelo del brazo robótico ha sido sacado de la documentación del robot YouBot y sólo ha sido necesario hacer unas pequeñas modificaciones para visualizarlo en el simulador Gazebo. Este modelo es controlable mediante ciertos topics de ROS, el los que se le puede indicar las posiciones de cada articulación del brazo. En este ejemplo el nodo procesador se encargará de descomponer los cuaterniones de orientación de los sensores en ángulos de Euler. Siendo específicos, se realizará una descomposición del cuaternión del sensor del brazo en los ángulos asociados a los ejes ZYX en dicho orden y en coordenadas intrínsecas mediante el algoritmo descrito en la memoria del proyecto. De estos ángulos se escoge el asociado al eje Z para la primera articulación y el asociado al eje Y para la segunda, desechando el asociado al eje X. Para la tercera articulación se utilizan los cuaterniones de brazo y antebrazo. Para esta articulación existe el problema del gimbal lock cuando los sensores se colocan a 90º uno con respecto del otro. Para eliminar este problema, lo que se hizo fue realizar una prerrotación de los sensores del brazo y antebrazo de 90º alrededor del eje local X de forma que desplazamos la zona problemática a un sitio donde sabemos que no se situarán los sensores. A continuación se aplica el mismo algoritmo, pero ahora se debe tener en cuenta que el ángulo que tenemos que escoger es el asociado al aje Z en lugar del Y. La cuarta articulación es idéntica a la tercera y se procederá de manera análoga. Una vez obtenidos los ángulos, se publica el mensaje para mover la simulación del brazo.

4.3. Tercera aplicación: Teleoperación del robot YouBot

La última aplicación creada es el parecida a la anterior, pero en lugar de controlar la simulación del brazo robótico, se controlará el brazo real del robot. Gracias a la modularidad de la que se a dotado al sistema se podrá usar el mismo nodo de procesamiento que en el ejemplo anterior. En este caso, al nodo actuador será el driver del robot YouBot, que interpretará los mismos topics que el simulador para mover el brazo robótico real. Por último, para demostrar la capacidad de ROS de permitir la ejecución de nodos en unidades de procesamiento distintas, los nodos sensor y procesador se ejecutarán en un ordenador portátil, y el nodo de actuación se ejecutará en el propio YouBot.

5. Conclusiones y trabajo futuro

Como conclusión se puede decir que se ha creado un sistema de monitorización fiable y de alta flexibilidad capaz de funcionar tanto en el cuerpo humano como en modelos de máquinas, en general en cualquier sistema articulado, en el que se ha aprovechando la gran cantidad de herramientas que proporciona la plataforma ROS. Por la condición de prototipo de este proyecto no se ha buscado un producto final cerrado, si no que se ha dejado la puerta abierta a la modificación e incorporación de otros sistemas de la forma más sencilla posible, y para facilitar esta tarea se han creado además varias librerías que facilitan enormemente la toma de datos de los sensores y la creación de aplicaciones para su procesamiento por otros desarrolladores. Además se ha implementado una arquitectura que posibilita la extensión del sistema incorporando un mayor número de sensores del mismo o distinto tipo, o incorporando todo tipo de sistemas actuadores que funcionen en la plataforma ROS. Y por último cabe mencionar que esta arquitectura además posibilita extender la ejecución de los distintos nodos a otros PCs y así aprovechar, si estuviera disponible, una mayor capacidad de procesamiento para tareas más exigentes computacionalmente.