

Entornos Operativos para Robótica e Informática Industrial (EOII)

Trabajo 2 – ROS2

Contenido

1.	Introducción	3
2.	Enunciado	3
3.	Pruebas	4
4.	Documentación a entregar	4
5.	Evaluación (sobre 10)	4

1. Introducción

El segundo trabajo de la asignatura tiene por objetivo evaluar al alumnado en los conocimientos vistos en las clases sobre ROS2 y en concreto:

1. Comandos del CLI de ROS2 Humble
2. Manejo de topics, servicios y action servers mediante aplicaciones Python
3. Paso de parámetros
4. Ficheros de launch

Para ello se estructura un trabajo con diferentes secciones, teniendo cada una de estas un peso propio en la nota final del trabajo.

2. Enunciado

Se requiere desarrollar un sistema completo de seguimiento de tortugas en el simulador TurtleSim que cumpla con las siguientes especificaciones:

E1-Generación de Tortuga Perseguidora

- El sistema debe generar automáticamente una nueva tortuga llamada "explorer" al iniciar
- La tortuga "explorer" debe aparecer en la posición (x=2.0, y=2.0) por defecto
- Se debe utilizar el servicio /spawn de TurtleSim

E2-Sistema de Seguimiento

- La tortuga "explorer" debe perseguir continuamente a la tortuga principal "turtle1" cuando esta es operada con la aplicación de teleoperación
- El algoritmo de seguimiento debe implementar control proporcional
- La velocidad debe ser proporcional a la distancia entre ambas tortugas

E3-Servicio de Información

- Implementar un servicio llamado turtle_info
- El servicio debe proporcionar:
 - Posición (x, y) de ambas tortugas
 - Orientación (theta) de ambas tortugas
 - Velocidades lineales y angulares actuales
 - Distancia euclidiana entre ambas tortugas

E4-Cliente de Consulta

- Desarrollar un nodo cliente que invoque el servicio turtle_info cada segundo
- El cliente debe mostrar la información de forma clara y estructurada
- Debe manejar errores si el servicio no está disponible

E5-Fichero launch

- Crear el fichero launch.xml que ponga en marcha el sistema
- Parametrizar el nodo servidor para que la posición de la tortuga explorer se indique como parámetro en el lanzamiento, comprobando que se encuentra en los límites del espacio de simulación

E6-Action Server

- Convertir el servicio de información en un Action Server que una vez pedido el servicio y aceptado, retorne periódicamente la información del E3. Se considerará terminado el servicio de información cuando la tortuga explorer, alcance a la tortuga turtle1 y se detenga

3. Pruebas

- Iniciar todos los nodos con el fichero de launch
- Usar `turtle_teleop_key` para mover `turtle1`
- Observar comportamiento de la tortuga `explorer`
- Mostrar en la consola los mensajes de información obtenidos por el cliente del servicio `turtle_info`

4. Documentación a entregar

1. **Código fuente:**
 - Fichero Python del nodo servidor
 - Fichero Python del nodo cliente
 - Comentarios explicativos en el código
2. **Memoria (PDF) (2-3 páginas):**
 - Descripción de la implementación
 - Resultados de pruebas
 - Problemas encontrados y soluciones
 - Descripción de los interfaces implementados así como los mensajes y servicios
 - Mapa de nodos, topics y servicios obtenido de `rqt_grap`

5. Evaluación

Sobre 10 puntos el peso de cada especificación que se cumpla será:

- E1: 1p
- E2: 2p
- E3 y E4: 2p
- E5: 1p
- E6: 2p
- Documentación: 2p
 - Código limpio, comentado y bien estructurado
 - Documentación completa de pruebas realizadas
 - Claridad y completitud de la documentación