Modelado y Simulación de Robots - GIRS

Práctica 3: Simulación de Robots usando middleware

Anexo: Lanzador los controladores

En este documento se muestran fragmentos de código para definir el fichero `*.launch.py` para lanzarlos controladores del robot.

Librerías y función principal

Se cargan las librerías y se define la función principal del *launch*. Además se establece la variable de entorno `use_sim_time` y la ruta del paquete donde se define el robot.

```
from os.path import join
from ament_index_python.packages import get_package_share_directory

from launch import LaunchDescription
from launch.actions import GroupAction, DeclareLaunchArgument
from controller_manager.launch_utils import generate_load_controller_launch_de

def generate_launch_description():
    declare_sim_time = DeclareLaunchArgument(
        'use_sim_time', default_value='true',
        description="use_sim_time simulation parameter"
    )

    pkg_share_folder = get_package_share_directory('msr_robot')
```

Publicador de estado y controlador de la base

El grupo `joint_state_broadcaster` se encarga de publicar los estados actuales (posición, velocidad, esfuerzo) de las articulaciones del robot en el tópico `/joint_states`. Además se añade el grupo de control de la base del robot. Ambos sistemas se encuentran

definidos en el fichero de configuración de los controladores (Ver documento *Ejemplo fichero de configuración de los controladores*).

```
# Load joint state broadcaster controller
joint_state_broadcaster = GroupAction(
    [
        generate_load_controller_launch_description(
            controller_name='joint_state_broadcaster',
            controller_params_file=join(
                pkg_share_folder, 'config', 'rover_controllers.yaml'))
    ],
)

# Load rover controller
base_controller = GroupAction(
    [
        generate_load_controller_launch_description(
            controller_name='rover_base_control',
            controller_params_file=join(
                pkg_share_folder, 'config', 'rover_controllers.yaml')
        )
    ],
)
```

Controladores del brazo y el efector final

Se añaden los controladores definidos en el paquete de configuración de la manipulación, generados previamente con el *MoveIt Setup Assistant*. En este caso, el archivo de configuración utilizado es el generado automáticamente por dicha herramienta, denominado por defecto `ros2_controllers.yaml` .

```
arm_pkg_share_folder = get_package_share_directory('rover_moveit_config')

# Load arm controller
arm_controller = GroupAction(
    [
        generate_load_controller_launch_description(
            controller_name='scara_controller',
            controller_params_file=join(
                arm_pkg_share_folder, 'config', 'ros2_controllers.yaml')
        )
    ],
)

# Load gripper controller
gripper_controller = GroupAction(
    [
        generate_load_controller_launch_description(
            controller_name='gripper_controller',
            controller_params_file=join(
                arm_pkg_share_folder, 'config', 'ros2_controllers.yaml')
        )
    ],
)
```

Llamada de acciones a ejecutar

Finalmente, para cada objeto creado se genera una acción de ejecución.

```
ld = LaunchDescription()
ld.add_action(joint_state_broadcaster)
ld.add_action(base_controller)
ld.add_action(arm_controller)
ld.add_action(gripper_controller)
ld.add_action(declare_sim_time)
return ld
```