Using MoveIt2

Planning robot motion with Moveit2

Jan van Hulzen

October 2, 2025

Overview

Assignment 1: Installation

- Check if Rviz2 is installed:

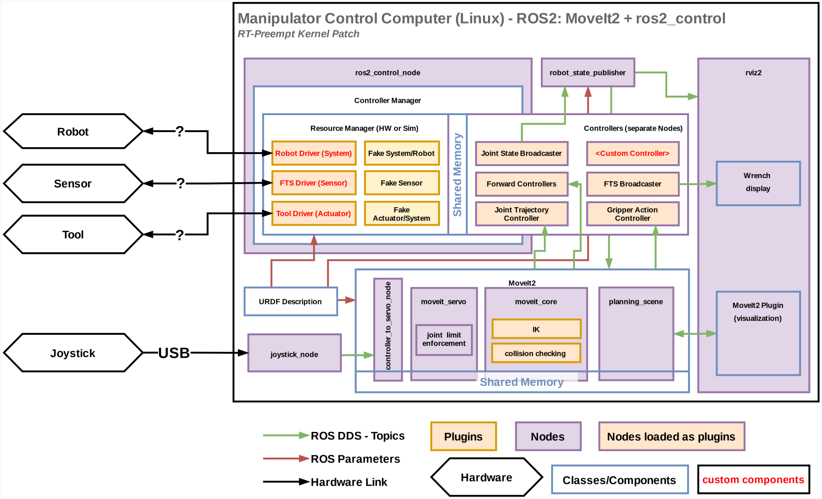
```
ros2 run rviz2 rviz2
```

- Install moveit2:
  https://moveit.ai/install-moveit2/binary/
- Check installation:

```
ros2 launch moveit2_tutorials demo.launch.py
```

- Bore-out preventer: https://moveit.picknik.ai/main/doc/tutorials/quickstart_in_rviz/quickstart_in_rviz_tutorial.html#getting-started

Overview software architecture ROS2 Control



Overview software architecture ROS2 Control

- Robot control is based on Moveit2+ros2_control
- Ros2_control nodes require configuration from .YAML files
  - Controller Manager
  - Resource Manager
  - Controllers (Joint State Breadcaster/Joint Trajectory Controller)
- Moveit planners require configuration
  - URDF
  - Moveit setup assistant to generate SRDF and .YAML files

Moveit2 setup Assistant

Assignment 2: Create package

- Starting from the results of previous lesson...
- Create directory structure:

```
cd ~/minor_ws/src
ros2 pkg create robot_moveit_config
cd robot_moveit_config
rm -r include/ src/
mkdir launch config
cd ..
code .
```

- Edit CMakeLists.txt

```
code .
```

Assignment 3: edit CMakeLists.txt

```cmake
cmake_minimum_required(VERSION 3.8)
project(robot_moveit_config)

find_package(ament_cmake REQUIRED)

# Install directories
install(
  DIRECTORY launch meshes urdf rviz
  DESTINATION share/${PROJECT_NAME}
)

ament_package()
```

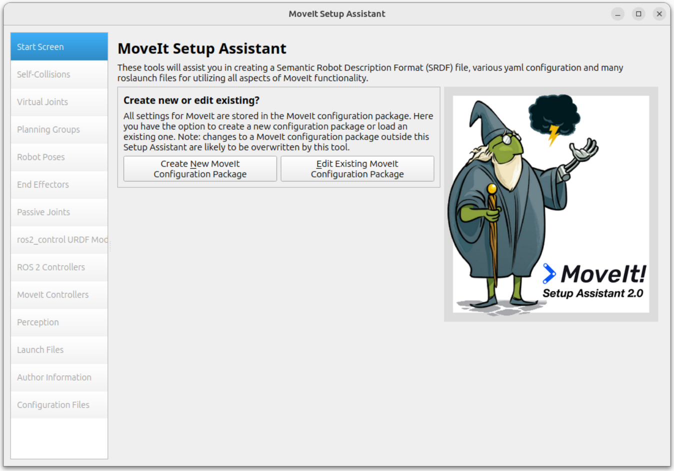
- Remove testing and add instruction to copy directories.
- Do this in package.xml as well.

Assignment 4: edit package.xml

```xml
<?xml version="1.0"?>
<?xml-model href="http:..."?>
<package format="3">
  <name>robot_moveit_config</name>
  <version>0.0.0</version>
  <description>Skyentific Robot Description package</
      description>
  <maintainer email="aap@noot.nl">Mies</maintainer>
  <license>Apache 2.0</license>
  <buildtool_depend>ament_cmake</buildtool_depend>
  <export>
    <build_type>ament_cmake</build_type>
  </export>
</package>
```

- Test build process with colcon build and source workspace...

Assignment 5: Moveit Setup Assistant



MoveIt Setup Assistant

These tools will assist you in creating a Semantic Robot Description Format (SRDF) file, various yaml configuration and many roslaunch files for utilizing all aspects of MoveIt functionality.

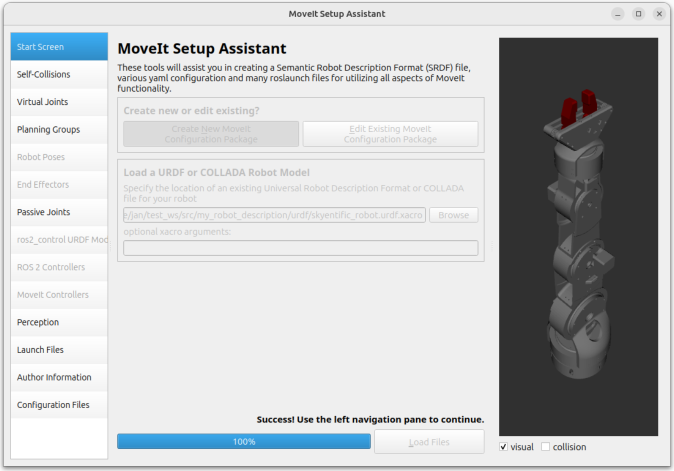
Create new or edit existing?

All settings for MoveIt are stored in the MoveIt configuration package. Here you have the option to create a new configuration package or load an existing one. Note: changes to a MoveIt configuration package outside this Setup Assistant are likely to be overwritten by this tool.

Create New MoveIt Configuration Package
Edit Existing MoveIt Configuration Package
MoveIt!
Setup Assistant 2.0

```
ros2 launch moveit_setup_assistant
    setup_assistant.launch.py
```

Assignment 6: Load files in Start Screen



MoveIt Setup Assistant

These tools will assist you in creating a Semantic Robot Description Format (SRDF) file, various yaml configuration and many roslaunch files for utilizing all aspects of MoveIt functionality.

Create new or edit existing?

Create New MoveIt Configuration Package
Edit Existing MoveIt Configuration Package

Load a URDF or COLLADA Robot Model

Specify the location of an existing Universal Robot Description Format or COLLADA file for your robot

e/jan/test_ws/src/my_robot_description/urdf/skyentific_robot.urdf.xacro

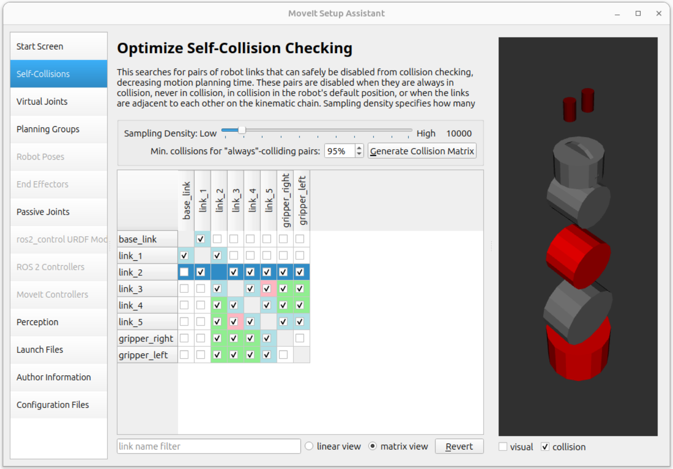
optional xacro arguments:

Success! Use the left navigation pane to continue.

✓ visual ☐ collision

- Select skyentific_robot.urdf.xacro in URDF directory.
- Load files and check robot in the panel on the right.

Assignment 7: Set Self-Collisions



MoveIt Setup Assistant

Optimize Self-Collision Checking

This searches for pairs of robot links that can safely be disabled from collision checking, decreasing motion planning time. These pairs are disabled when they are always in collision, never in collision, in collision in the robot's default position, or when the links are adjacent to each other on the kinematic chain. Sampling density specifies how many

Sampling Density: Low High 10000

Min. collisions for "always"-colliding pairs: 95%

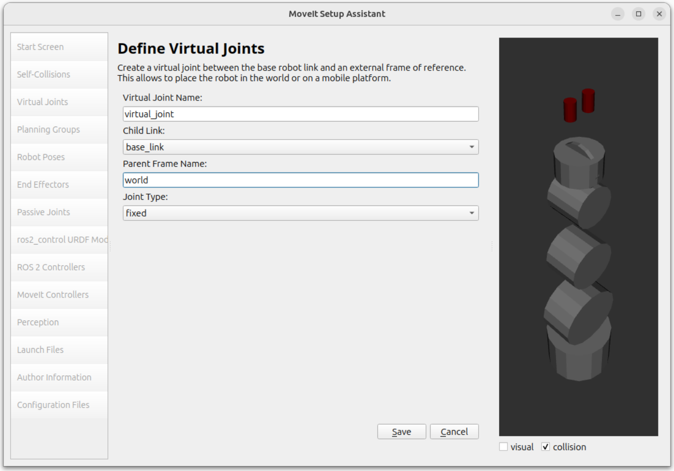
	base_link	link_1	link_2	link_3	link_4	link_5	gripper_right	gripper_left
base_link		✓	☐	☐	☐	☐	☐	☐
link_1	✓		✓	☐	☐	☐	☐	☐
link_2	☐	✓		✓	✓	✓	✓	✓
link_3	☐	☐	✓		✓	✓	✓	✓
link_4	☐	☐	✓	✓		✓	✓	✓
link_5	☐	☐	✓	✓	✓		✓	✓
gripper_right	☐	☐	✓	✓	✓	✓		☐
gripper_left	☐	☐	✓	✓	✓	✓	☐	

link name filter ○ linear view [● matrix view]

☐ visual ✓ collision

- Select Collisions on te right an generate Collision Matrix.
- Check Collisions and exclude adjacent links or links that are never in contact from collision checking.

Assignment 8: Add virtual joint



Define Virtual Joints

Create a virtual joint between the base robot link and an external frame of reference.
This allows to place the robot in the world or on a mobile platform.

Virtual Joint Name: virtual_joint

Child Link: base_link

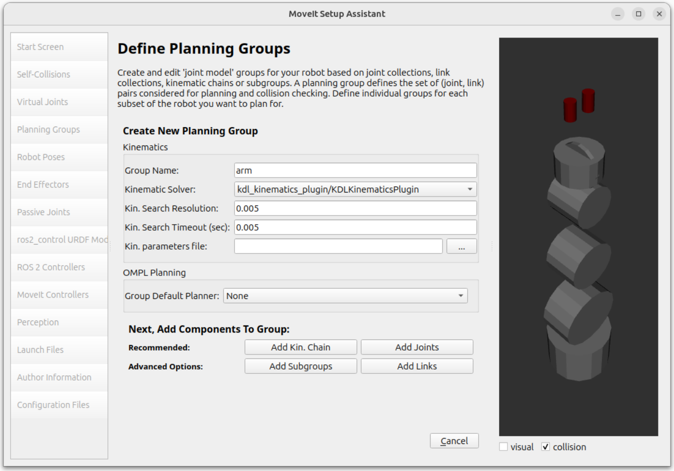
Parent Frame Name: world

Joint Type: fixed

☐ visual ☑ collision

- Add a virtual joint of type fixed to anchor robot.
- Use world as parent and base_link as child.

Assignment 9: Define planning group arm



Define Planning Groups

Create and edit 'joint model' groups for your robot based on joint collections, link collections, kinematic chains or subgroups. A planning group defines the set of (joint, link) pairs considered for planning and collision checking. Define individual groups for each subset of the robot you want to plan for.

Create New Planning Group

Kinematics

Group Name:	arm
Kinematic Solver:	kdl_kinematics_plugin/KDLKinematicsPlugin
Kin. Search Resolution:	0.005
Kin. Search Timeout (sec):	0.005
Kin. parameters file:	

OMPL Planning

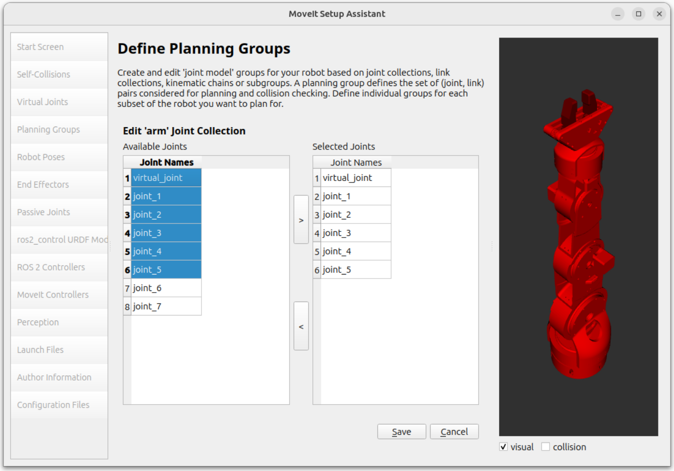
Group Default Planner: None

Next, Add Components To Group:

Recommended: Add Kin. Chain, Add Joints
Advanced Options: Add Subgroups, Add Links
Cancel
visual
collision

- Add a planning group and select Add Joints.
- Select kdl_kinematics_plugin/KDLKinematicsPlugin.

Assignment 10: Add joints to group arm



Define Planning Groups

Create and edit 'joint model' groups for your robot based on joint collections, link collections, kinematic chains or subgroups. A planning group defines the set of (joint, link) pairs considered for planning and collision checking. Define individual groups for each subset of the robot you want to plan for.

Edit 'arm' Joint Collection

Available Joints

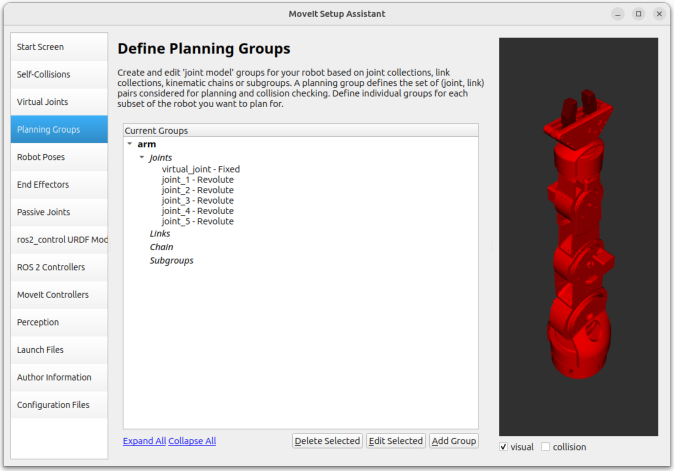
	Joint Names
1	virtual_joint
2	joint_1
3	joint_2
4	joint_3
5	joint_4
6	joint_5
7	joint_6
8	joint_7

Selected Joints

	Joint Names
1	virtual_joint
2	joint_1
3	joint_2
4	joint_3
5	joint_4
6	joint_5

- Add virtual joint and joints 1-5 to group.
- Joint_6 and joint_7 will be in the gripper group.

Assignment 11: Check settings for group arm



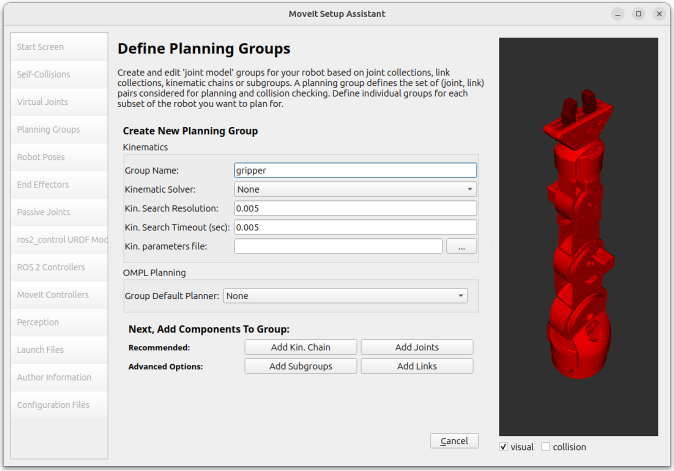
Define Planning Groups

Create and edit 'joint model' groups for your robot based on joint collections, link collections, kinematic chains or subgroups. A planning group defines the set of (joint, link) pairs considered for planning and collision checking. Define individual groups for each subset of the robot you want to plan for.

Current Groups
- **arm**
  - *Joints*
    - virtual_joint - Fixed
    - joint_1 - Revolute
    - joint_2 - Revolute
    - joint_3 - Revolute
    - joint_4 - Revolute
    - joint_5 - Revolute
  - *Links*
  - *Chain*
  - *Subgroups*

- All joints of arm should be of type revolute.
- Links, Chain and Subgroups are empty.

Assignment 12: Define planning group gripper



Define Planning Groups

Create and edit 'joint model' groups for your robot based on joint collections, link collections, kinematic chains or subgroups. A planning group defines the set of (joint, link) pairs considered for planning and collision checking. Define individual groups for each subset of the robot you want to plan for.

Create New Planning Group

Kinematics

Group Name:	gripper
Kinematic Solver:	None
Kin. Search Resolution:	0.005
Kin. Search Timeout (sec):	0.005
Kin. parameters file:	

OMPL Planning

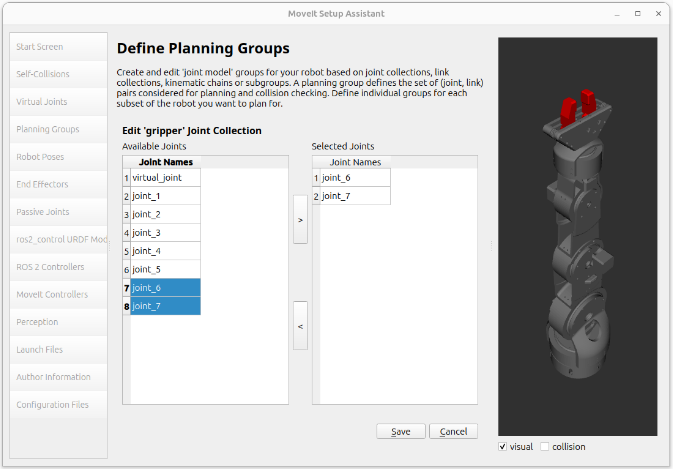
Group Default Planner: None

Next, Add Components To Group:

Recommended: Add Kin. Chain Add Joints
Advanced Options: Add Subgroups Add Links
Cancel
visual
collision

- The name of the group should be gripper.
- Select None as kinmetic solver type.

Assignment 13: Add joints to group gripper



Define Planning Groups

Create and edit 'joint model' groups for your robot based on joint collections, link collections, kinematic chains or subgroups. A planning group defines the set of (joint, link) pairs considered for planning and collision checking. Define individual groups for each subset of the robot you want to plan for.

Edit 'gripper' Joint Collection

Available Joints

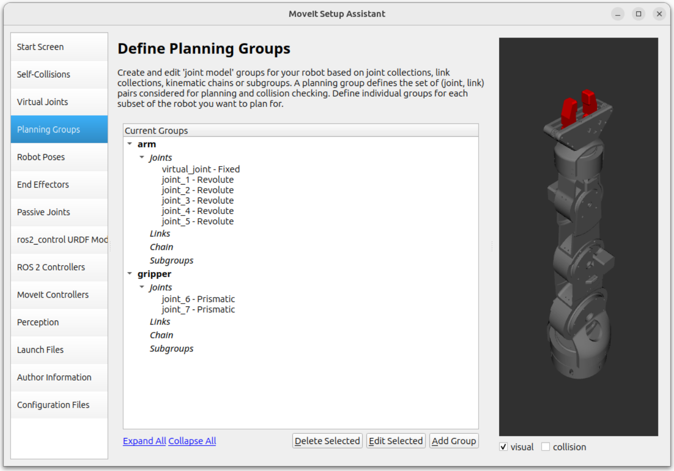
	Joint Names
1	virtual_joint
2	joint_1
3	joint_2
4	joint_3
5	joint_4
6	joint_5
7	joint_6
8	joint_7

Selected Joints

	Joint Names
1	joint_6
2	joint_7

- Add joint_6 and joint_7 to group.

Assignment 14: Check settings for group gripper



Define Planning Groups

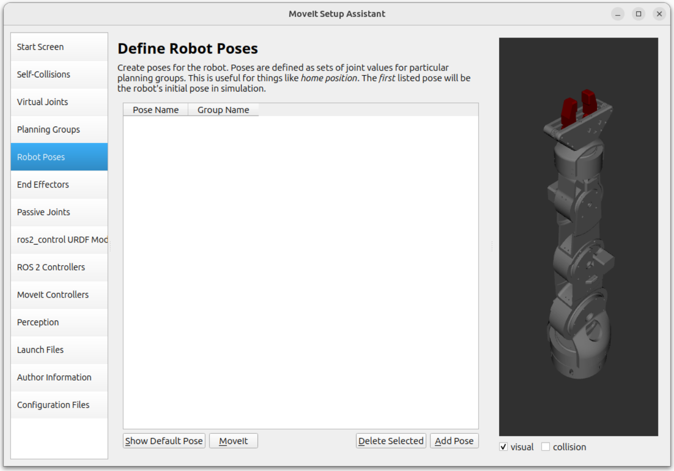
Create and edit 'joint model' groups for your robot based on joint collections, link collections, kinematic chains or subgroups. A planning group defines the set of (joint, link) pairs considered for planning and collision checking. Define individual groups for each subset of the robot you want to plan for.

```
Current Groups
▾ arm
  ▾ Joints
      virtual_joint - Fixed
      joint_1 - Revolute
      joint_2 - Revolute
      joint_3 - Revolute
      joint_4 - Revolute
      joint_5 - Revolute
    Links
    Chain
    Subgroups
▾ gripper
  ▾ Joints
      joint_6 - Prismatic
      joint_7 - Prismatic
    Links
    Chain
    Subgroups
```

[✓ visual] collision

- Joint_6 and joint_7 should be of type prismatic.

Assignment 15: Define robot poses

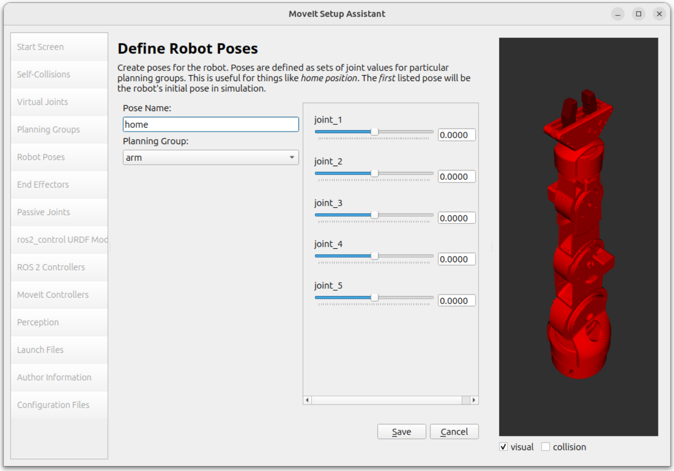


Define Robot Poses

Create poses for the robot. Poses are defined as sets of joint values for particular planning groups. This is useful for things like *home position*. The *first* listed pose will be the robot's initial pose in simulation.

- Robot poses can be set at this stage or later in code.
- Define at least one pose for each group and edit this later.

Assignment 16: Define home postion for group arm



Define Robot Poses

Create poses for the robot. Poses are defined as sets of joint values for particular planning groups. This is useful for things like *home position*. The *first* listed pose will be the robot's initial pose in simulation.

Pose Name: home

Planning Group: arm

joint_1 0.0000

joint_2 0.0000

joint_3 0.0000

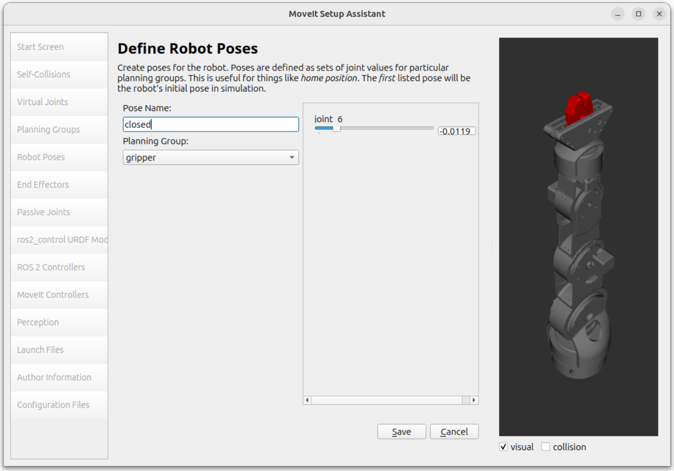
joint_4 0.0000

joint_5 0.0000

☑ visual ☐ collision

- Defining a home position with all joints set to zero.

Assignment 17: Define home postions for group gripper


MoveIt Setup Assistant

Define Robot Poses

Create poses for the robot. Poses are defined as sets of joint values for particular planning groups. This is useful for things like *home position*. The *first* listed pose will be the robot's initial pose in simulation.

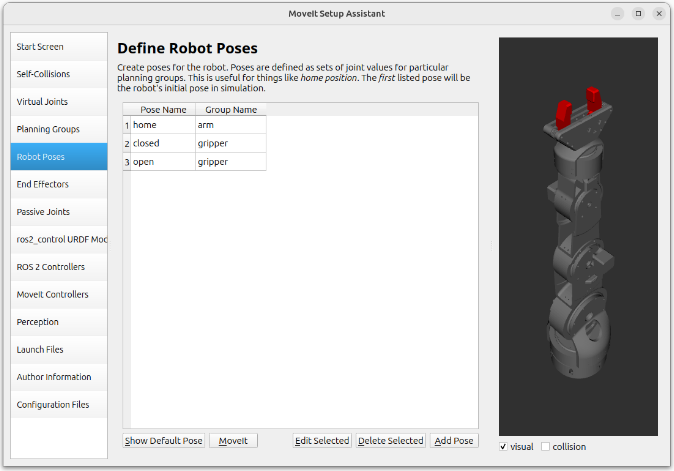
Pose Name: closed

Planning Group: gripper

joint_6 -0.0119

- Group gripper has only one slider since joint_7 mimics joint_6.
- Define appropriate setting for posea open and closed.

Assignment 18: Check resulting poses



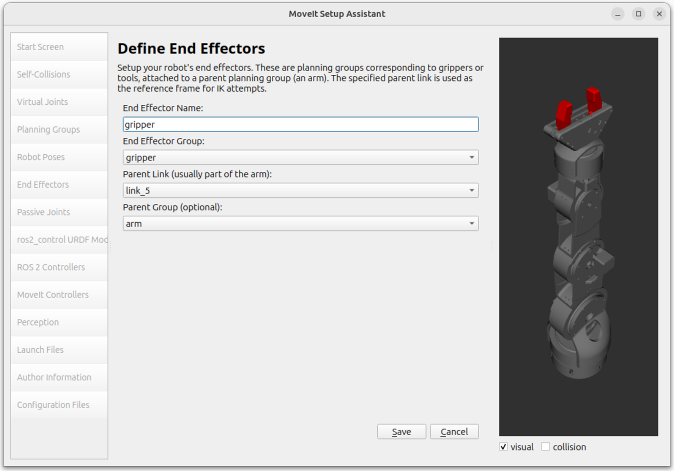
Define Robot Poses

Create poses for the robot. Poses are defined as sets of joint values for particular planning groups. This is useful for things like *home position*. The *first* listed pose will be the robot's initial pose in simulation.

	Pose Name	Group Name
1	home	arm
2	closed	gripper
3	open	gripper

- check results

Assignment 19: Define end effectors


MoveIt Setup Assistant

Define End Effectors

Setup your robot's end effectors. These are planning groups corresponding to grippers or tools, attached to a parent planning group (an arm). The specified parent link is used as the reference frame for IK attempts.

End Effector Name: gripper

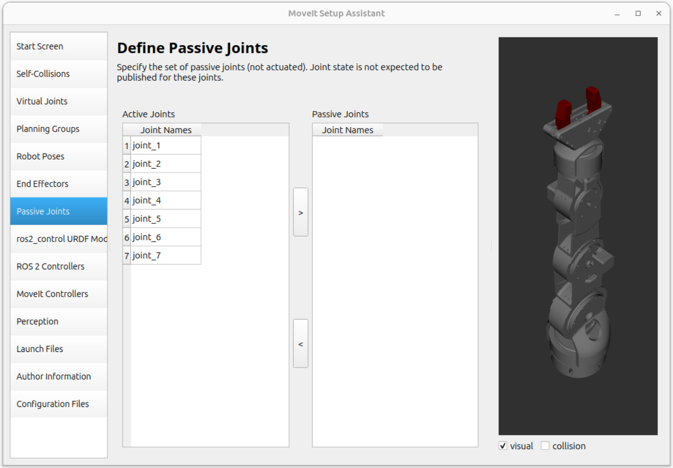
End Effector Group: gripper

Parent Link (usually part of the arm): link_5

Parent Group (optional): arm

- set end effector group to gripper.
- select link_5 as parent and arm a parent group.

Assignment 20: Define passive joints



Define Passive Joints

Specify the set of passive joints (not actuated). Joint state is not expected to be published for these joints.

Active Joints

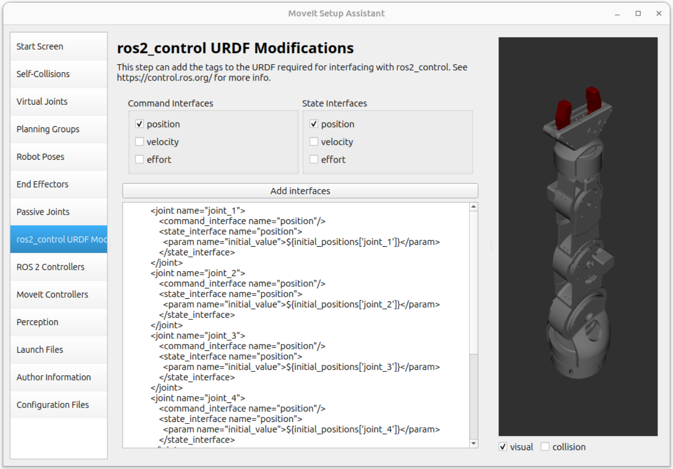
	Joint Names
1	joint_1
2	joint_2
3	joint_3
4	joint_4
5	joint_5
6	joint_6
7	joint_7

Passive Joints

Joint Names

- We have no passive joints, so leave this blank.

Assignment 21: Define ros2_control URDF Modifications



ros2_control URDF Modifications

This step can add the tags to the URDF required for interfacing with ros2_control. See https://control.ros.org/ for more info.

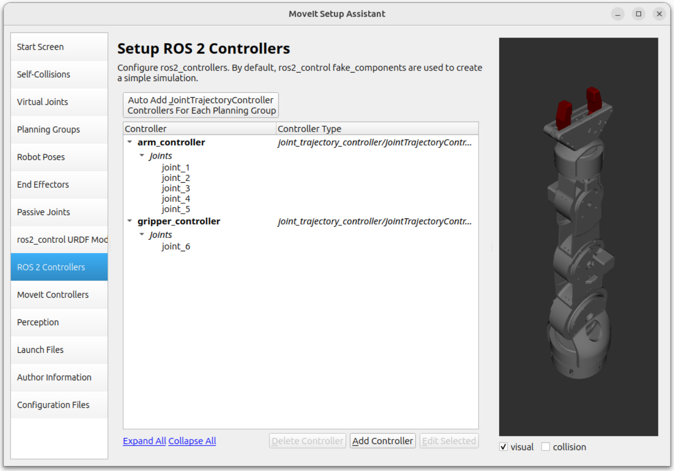
Command Interfaces
[✓ position]
velocity
effort

State Interfaces
[✓ position]
velocity
effort

```
<joint name="joint_1">
  <command_interface name="position"/>
  <state_interface name="position">
    <param name="initial_value">${initial_positions['joint_1']}</param>
  </state_interface>
</joint>
<joint name="joint_2">
  <command_interface name="position"/>
  <state_interface name="position">
    <param name="initial_value">${initial_positions['joint_2']}</param>
  </state_interface>
</joint>
<joint name="joint_3">
  <command_interface name="position"/>
  <state_interface name="position">
    <param name="initial_value">${initial_positions['joint_3']}</param>
  </state_interface>
</joint>
<joint name="joint_4">
  <command_interface name="position"/>
  <state_interface name="position">
    <param name="initial_value">${initial_positions['joint_4']}</param>
  </state_interface>
```

- Select position for command and state interfaces.
- We need to press the Add interfaces button for the changes to have effect.

Assignment 22: Setup ROS2 Controllers



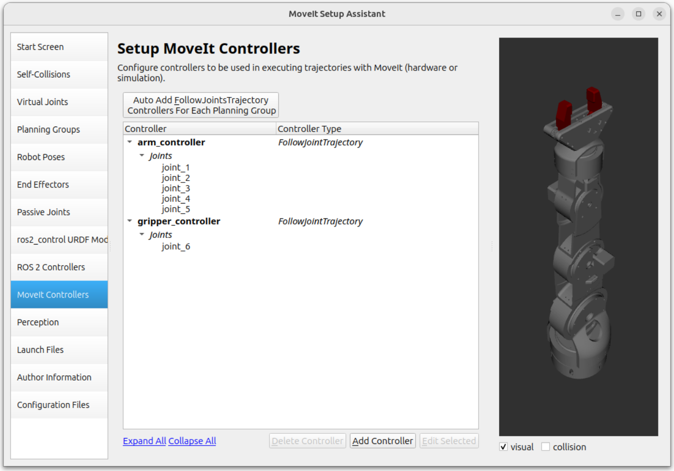
Setup ROS 2 Controllers

Configure ros2_controllers. By default, ros2_control fake_components are used to create a simple simulation.

Controller	Controller Type
arm_controller	*joint_trajectory_controller/JointTrajectoryContr...*
Joints	
joint_1	
joint_2	
joint_3	
joint_4	
joint_5	
gripper_controller	*joint_trajectory_controller/JointTrajectoryContr...*
Joints	
joint_6	

- Use Auto Add JointTrajectoryController.

Assignment 23: Setup MoveIt Controllers



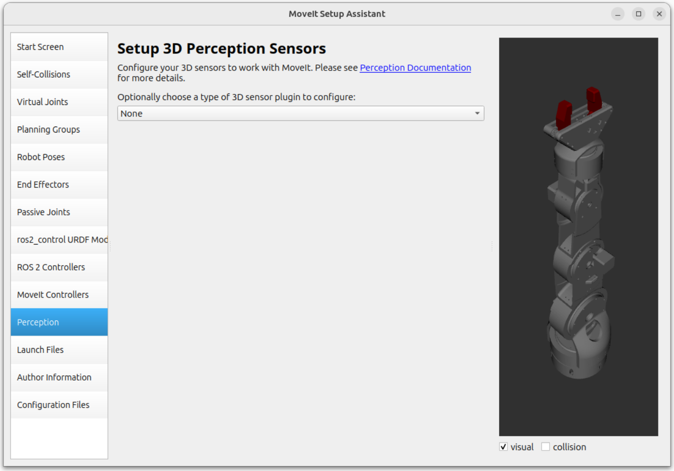
Setup MoveIt Controllers

Configure controllers to be used in executing trajectories with MoveIt (hardware or simulation).

Controller	Controller Type
arm_controller	*FollowJointTrajectory*
Joints	
joint_1	
joint_2	
joint_3	
joint_4	
joint_5	
gripper_controller	*FollowJointTrajectory*
Joints	
joint_6	

- Use Auto Add FollowjointTrajectory Controllers for each planning group..

Assignment 24: Setup 3D perseption Sensors



Setup 3D Perception Sensors

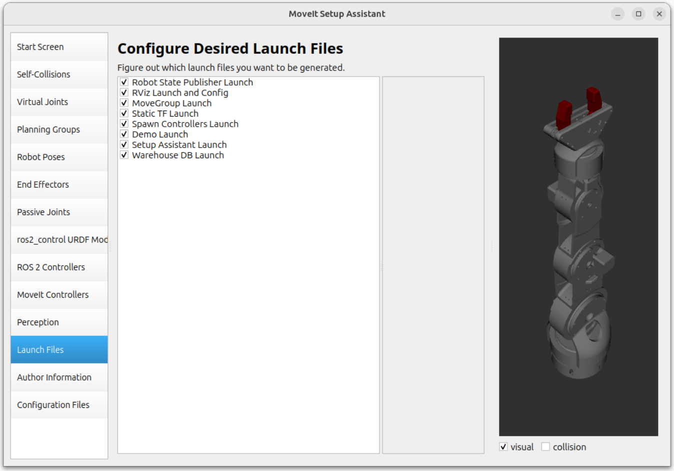
Configure your 3D sensors to work with MoveIt. Please see Perception Documentation for more details.

Optionally choose a type of 3D sensor plugin to configure:

None

- Leave this blank.

Assignment 25: Select Launch Files



Configure Desired Launch Files

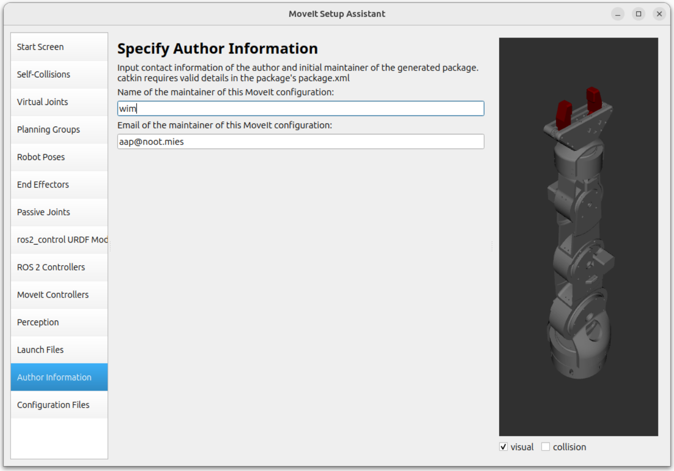
Figure out which launch files you want to be generated.

- [x] Robot State Publisher Launch
- [x] RViz Launch and Config
- [x] MoveGroup Launch
- [x] Static TF Launch
- [x] Spawn Controllers Launch
- [x] Demo Launch
- [x] Setup Assistant Launch
- [x] Warehouse DB Launch

[✓ visual] collision

- Select all, why don't ya.

Assignment 26: Specify author informaton



Specify Author Information

Input contact information of the author and initial maintainer of the generated package. catkin requires valid details in the package's package.xml

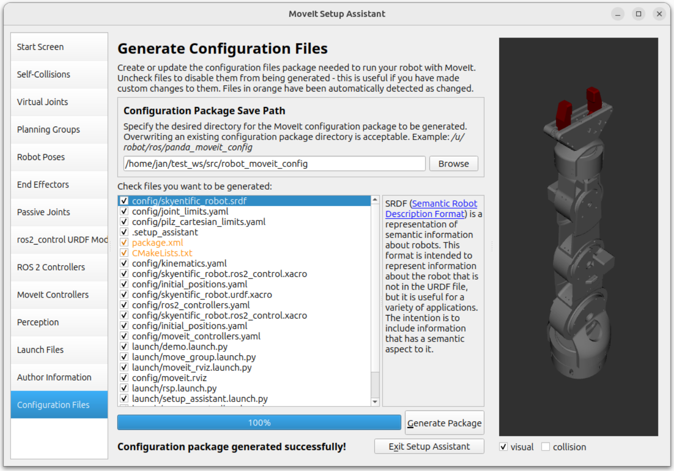
Name of the maintainer of this MoveIt configuration: wim

Email of the maintainer of this MoveIt configuration: aap@noot.mies

☑ visual ☐ collision

- Do not leave this blank.

Assignment 27: Generate configuration file.



Generate Configuration Files

Create or update the configuration files package needed to run your robot with MoveIt. Uncheck files to disable them from being generated - this is useful if you have made custom changes to them. Files in orange have been automatically detected as changed.

Configuration Package Save Path

Specify the desired directory for the MoveIt configuration package to be generated. Overwriting an existing configuration package directory is acceptable. Example: */u/robot/ros/panda_moveit_config*

/home/jan/test_ws/src/robot_moveit_config

Check files you want to be generated:

- ✓ config/skyentific_robot.srdf
- ✓ config/joint_limits.yaml
- ✓ config/pilz_cartesian_limits.yaml
- ✓ .setup_assistant
- ✓ package.xml
- ✓ CMakeLists.txt
- ✓ config/kinematics.yaml
- ✓ config/skyentific_robot.ros2_control.xacro
- ✓ config/initial_positions.yaml
- ✓ config/skyentific_robot.urdf.xacro
- ✓ config/ros2_controllers.yaml
- ✓ config/skyentific_robot.ros2_control.xacro
- ✓ config/initial_positions.yaml
- ✓ config/moveit_controllers.yaml
- ✓ launch/demo.launch.py
- ✓ launch/move_group.launch.py
- ✓ launch/moveit_rviz.launch.py
- ✓ config/moveit.rviz
- ✓ launch/rsp.launch.py
- ✓ launch/setup_assistant.launch.py

SRDF (Semantic Robot Description Format) is a representation of semantic information about robots. This format is intended to represent information about the robot that is not in the URDF file, but it is useful for a variety of applications. The intention is to include information that has a semantic aspect to it.

100%

Configuration package generated successfully!

- Select the appropriate package in your workspace.
- If you get an error about the .setup_assistant file just create an empy one using a terminal and try again.

Assignment 28: Explore initial_positions.yaml

```
robot_moveit_config
└─config
  └─initial_positions.yaml
```

```
# Default initial positions for skyentific_robot's
    ros2_control fake system

initial_positions:
  joint_1: 0
  joint_2: 0
  joint_3: 0
  joint_4: 0
  joint_5: 0
  joint_6: 0
```

Assignment 29: Edit file joint_limits.yaml

```
robot_moveit_config
└ config
  └ joint_limits.yaml
```

```
# Allows the dynamics properties specified in the URDF
    to be overwritten
default_velocity_scaling_factor: 0.1  # Scaled to 10%
default_acceleration_scaling_factor: 0.1 # for safety
joint_limits:
  joint_1:
    has_velocity_limits: true
    max_velocity: 10.0 # use type double
    has_acceleration_limits: true # set to true
    max_acceleration: 10.0 # use type double
  # change this for all joints...
```

Assignment 30: Expore kinematics.yaml

```
robot_moveit_config
└─config
  └─kinematics.yaml
```

```
arm:
  kinematics_solver:
      kdl_kinematics_plugin/KDLKinematicsPlugin
  kinematics_solver_search_resolution:
      0.005000000000000001
  kinematics_solver_timeout: 0.005000000000000001
```

Assignment 30: Edit file moveit_controllers.yaml 1/3

```
robot_moveit_config
└ config
  └ moveit_controllers.yaml
```

```yaml
# MoveIt uses this config for controller management

moveit_controller_manager:
    moveit_simple_controller_manager/
            MoveItSimpleControllerManager

moveit_simple_controller_manager:
  controller_names:
    - arm_controller
    - gripper_controller
```

Assignment 30: Edit file moveit_controllers.yaml 2/3

```yaml
arm_controller:
  type: FollowJointTrajectory
  joints:
    - joint_1
    - joint_2
    - joint_3
    - joint_4
    - joint_5
gripper_controller:
  type: FollowJointTrajectory
  joints:
    - joint_6
```

- We need to make some changes here
- Add action_ns: follow_joint_trajectory
- Add default: true

Assignment 30: Edit file moveit_controllers.yaml 3/3

```yaml
arm_controller:
  type: FollowJointTrajectory
  joints:
    - joint_1
    - joint_2
    - joint_3
    - joint_4
    - joint_5
  action_ns: follow_joint_trajectory
  default: true
gripper_controller:
  type: FollowJointTrajectory
  joints:
    - joint_6
  action_ns: follow_joint_trajectory
  default: true
```

Assignment 31: Check file moveit.rviz

```
robot_moveit_config
└ config
   └ moveit.rviz
```

```
Panels:
  - Class: rviz_common/Displays
    Name: Displays
    Property Tree Widget:
      Expanded:
        - /MotionPlanning1
  - Class: rviz_common/Help
    Name: Help
  - Class: rviz_common/Views
    Name: Views
Visualization Manager:
  Displays:
    - Class: rviz_default_plugins/Grid
```

Assignment 32: Explore pilz_cartesian_limits.yaml

```
robot_moveit_config
└ config
   └ pilz_cartesian_limits.yaml
```

```yaml
# Limits for the Pilz planner
cartesian_limits:
  max_trans_vel: 1.0
  max_trans_acc: 2.25
  max_trans_dec: -5.0
  max_rot_vel: 1.57
```

Assignment 33: Edit file ros2_controllers.yaml 1/2

```
robot_moveit_config
└ config
  └ ros2_controllers.yaml
```

```
controller_manager: # config file used by ros2_control
  ros__parameters:
    update_rate: 1 # Hz (Limit this a bit)
    arm_controller:
      type: joint_trajectory_controller/
                      JointTrajectoryController
    gripper_controller:
      type: joint_trajectory_controller/
                      JointTrajectoryController
    joint_state_broadcaster:
      type: joint_state_broadcaster/
                          JointStateBroadcaster
```

Assignment 34: Edit file ros2_controllers.yaml 2/2

```
arm_controller:
  ros__parameters:
    joints:
      - joint_1
        ...
    command_interfaces:
      - position
    state_interfaces:
      - position
    allow_nonzero_velocity_at_trajectory_end: true
gripper_controller:
  ros__parameters:
    joints:
      - joint_6
    command_interfaces:
      - position
    state_interfaces:
      - position
    allow_nonzero_velocity_at_trajectory_end: true
```

Assignment 35: Explore skyentific_robot.ros2_control.xacro

```
robot_moveit_config
└ config
   └ skyentific_robot.ros2_control.xacro
```

```
<?xml version="1.0"?>
<robot xmlns:xacro="http://www.ros.org/wiki/xacro">
   <xacro:macro name="skyentific_robot_ros2_control"
       params="name initial_positions_file">
   <xacro:property name="initial_positions" value="${
       xacro.load_yaml(initial_positions_file)
                               ['initial_positions']}"/>
     ...
  </xacro:macro>
</robot>
```

Assignment 36: Explore skyentific_robot.ros2_control.xacro

```
<ros2_control name="${name}" type="system">
 <hardware>
    <!-- By default, set up controllers for simulation.
        This won't work on real hardware -->
    <plugin>mock_components/GenericSystem</plugin>
 </hardware>
 <joint name="joint_1">
    <command_interface name="position"/>
    <state_interface name="position">
     <param name="initial_value">
              ${initial_positions['joint_1']}</param>
    </state_interface>
 </joint>
 ...
</ros2_control>
```

- The xacro nested macro structure is used to reduce complexity.

Assignment 37: Explore skyentific_robot.srdf

```
robot_moveit_config
└ config
   └ skyentific_robot.srdf
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<robot name="skyentific_robot">
 <group name="arm">
  <joint name="virtual_joint"/>
  <joint name="joint_1"/>
  ...
 </group>
 <group name="gripper">
  <joint name="joint_6"/>
  <joint name="joint_7"/>
 </group>
 ...
```

Assignment 37: Explore skyentific_robot.srdf

```
<!--GROUP STATES: Purpose: Define a named state for a
    particular group, in terms of joint values. This is
    useful to define states like 'folded arms'-->
<group_state name="home" group="arm">
   <joint name="joint_1" value="0"/>
   <joint name="joint_2" value="0"/>
   <joint name="joint_3" value="0"/>
   <joint name="joint_4" value="0"/>
   <joint name="joint_5" value="0"/>
</group_state>
<group_state name="closed" group="gripper">
   <joint name="joint_6" value="-0.0119"/>
</group_state>
<group_state name="open" group="gripper">
   <joint name="joint_6" value="0.014"/>
</group_state>
```

- We will edit this section at a later stage.

Assignment 37: Explore skyentific_robot.srdf

```
<!--END EFFECTOR: Purpose: Represent information...-->
<end_effector name="gripper" parent_link="link_5" group=
    "gripper" parent_group="arm"/>
<!--VIRTUAL JOINT: Purpose: this element defines a
    virtual joint between a robot link and an external
    frame of reference (considered fixed )-->
<virtual_joint name="virtual_joint" type="fixed"
    parent_frame="world" child_link="base_link"/>
<!--DISABLE COLLISIONS: By default it is assumed that
    any link of the robot could potentially come into
    collision with any other link in the robot. This tag
    disables collision checking. -->
<disable_collisions link1="base_link" link2="link_1"
    reason="Adjacent"/>
<disable_collisions link1="gripper_left" link2="link_2"
    reason="Never"/>
...
```

Assignment 38: Explore package.xml 1/4

📁 robot_moveit_config

└ 📄 package.xml

```
<?xml version="1.0"?>
<?xml-model href="http://download.ros.org/schema/
package_format3.xsd" schematypens="http://www.w3.org
    /2001/XMLSchema"?>
<package format="3">
  <name>robot_moveit_config</name>
  <version>0.3.0</version>
  <description>
    An automatically generated package with all the
        configuration and launch files for using the
        robot with the MoveIt Motion Planning Framework
  </description>
```

Assignment 38: Explore package.xml 2/4

```
<maintainer email="jan.vanhulzen@inholland.nl">jan van
    hulzen</maintainer>
<license>BSD-3-Clause</license>
<url type="website">http://moveit.ros.org/</url>
<url type="bugtracker">https://github.com/moveit/
moveit2/issues</url>
<url type="repository">https://github.com/moveit/
moveit2</url>
<author email="jan.vanhulzen@inholland.nl">
jan van hulzen</author>
<buildtool_depend>ament_cmake</buildtool_depend>

<exec_depend>moveit_ros_move_group</exec_depend>
<exec_depend>moveit_kinematics</exec_depend>
<exec_depend>moveit_planners</exec_depend>
<exec_depend>moveit_simple_controller_manager
     </exec_depend>
```

Assignment 38: Explore package.xml 3/4

```xml
<exec_depend>joint_state_publisher</exec_depend>
<exec_depend>joint_state_publisher_gui</exec_depend>
<exec_depend>tf2_ros</exec_depend>
<exec_depend>xacro</exec_depend>
<exec_depend>controller_manager</exec_depend>
<exec_depend>moveit_configs_utils</exec_depend>
<exec_depend>moveit_ros_move_group</exec_depend>
<exec_depend>moveit_ros_visualization</exec_depend>
<exec_depend>moveit_ros_warehouse</exec_depend>
<exec_depend>moveit_setup_assistant</exec_depend>
<exec_depend>robot_description</exec_depend>
<exec_depend>robot_state_publisher</exec_depend>
```

Assignment 38: Explore package.xml 4/4

```
  <exec_depend>rviz2</exec_depend>
  <exec_depend>rviz_common</exec_depend>
  <exec_depend>rviz_default_plugins</exec_depend>
  <exec_depend>tf2_ros</exec_depend>
  <exec_depend>warehouse_ros_mongo</exec_depend>
  <exec_depend>xacro</exec_depend>

  <export>
    <build_type>ament_cmake</build_type>
  </export>
</package>
```

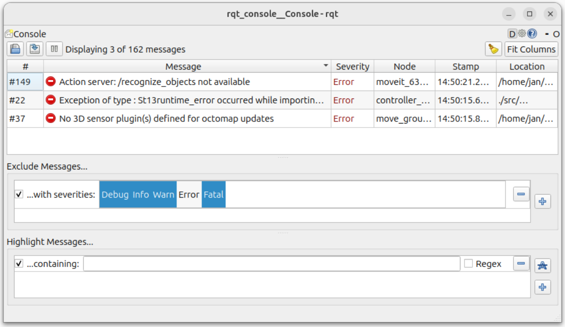
Assignment 39: Explore demo.launch.py

```
robot_moveit_config
└─ launch
   └─ demo.launch.py
```

```python
from moveit_configs_utils import
    MoveItConfigsBuilder
from moveit_configs_utils.launches import
    generate_demo_launch

def generate_launch_description():
 moveit_config = MoveItConfigsBuilder("
     skyentific_robot",
 package_name="robot_moveit_config")
     .to_moveit_configs()
 return generate_demo_launch(moveit_config)
```

Assignment 40: Test package


rqt_console__Console - rqt
Console
Displaying 3 of 162 messages
Fit Columns
Exclude Messages...
...with severities: Debug Info Warn Error Fatal
Highlight Messages...
...containing:
Regex

#	Message	Severity	Node	Stamp	Location
#149	Action server: /recognize_objects not available	Error	moveit_63...	14:50:21.2...	/home/jan/...
#22	Exception of type : St13runtime_error occurred while importin...	Error	controller_...	14:50:15.6...	./src/...
#37	No 3D sensor plugin(s) defined for octomap updates	Error	move_grou...	14:50:15.8...	/home/jan/...

- Build packages, start rqt_console, run package.

```
colcon build && source install/setup.bash
ros2 launch robot_moveit_config demo.launch.py
```

- Use a second terminal open rqt_console to check errors.

```
ros2 run rqt_console rqt_console
```

Defining a robot_bringup package

Assignment 41: Create package robot_bringup

- Build package:

```
cd ~/minor_ws/src
ros2 pkg create robot_bringup
cd robot_bringup
rm -r include/ src/
mkdir launch config
touch config/skyentific_robot_controllers.yaml
cd ~/minor_ws/
code .
```

Assignment 42: edit CMakeLists.txt

```
cmake_minimum_required(VERSION 3.8)
project(robot_bringup)

find_package(ament_cmake REQUIRED)

# Install directories
install(
  DIRECTORY launch config
  DESTINATION share/${PROJECT_NAME}
)

ament_package()
```

- Remove testing and add instruction to copy directories.
- No need to edit the package.xml.

Assignment 43: Define skyentific_robot_controllers.yaml

- We will add hardware parameters here at a later stage.

```yaml
controller_manager: # config file used by ros2_control
  ros__parameters:
    update_rate: 1 # Hz (Limit this a bit)
    arm_controller:
      type: joint_trajectory_controller/
                  JointTrajectoryController
    gripper_controller:
      type: joint_trajectory_controller/
                  JointTrajectoryController
    joint_state_broadcaster:
      type: joint_state_broadcaster/
                  JointStateBroadcaster
```

Assignment 43: Define skyentific_robot_controllers.yaml

```yaml
arm_controller:
  ros__parameters:
    joints:
      - joint_1
        ...
    command_interfaces:
      - position
    state_interfaces:
      - position
    allow_nonzero_velocity_at_trajectory_end: true
gripper_controller:
  ros__parameters:
    joints:
      - joint_6
    command_interfaces:
      - position
    state_interfaces:
      - position
    allow_nonzero_velocity_at_trajectory_end: true
```

Assignment 44: Start robot_state_publisher

- Start robot state publisher (type correctly!)

```
source install/setup.bash
ros2 run robot_state_publisher
   robot_state_publisher --ros-args -p
   robot_description:="$(xacro␣/home/jan/minor_ws/
   src/robot_description/urdf/
   skyentific_robot.urdf.xacro)"
```

- In a second terminal:

```
source install/setup.bash
ros2 node list
ros2 param list /robot_state_publisher
```

Assignment 45: Start controller_manager

- Start controller_manager (type correctly!)

```
source install/setup.bash
ros2 run controller_manager ros2_control_node --ros
   -args --params-file /home/jan/minor_ws/src/
   robot_bringup/config/
skyentific_robot_controllers.yaml
```

- In a second terminal:

```
source install/setup.bash
ros2 node list
ros2 param list /controller\_manager
```

Assignment 46: Start controllers

- Start controllers

```
source install/setup.bash
ros2 run controller_manager spawner
   joint_state_broadcaster
ros2 run controller_manager spawner arm_controller
ros2 run controller_manager spawner
   gripper_controller
```

- In a second terminal:

```
source install/setup.bash
ros2 node list
ros2 param list /controller\_manager
```

Assignment 47: Start moveit

- Start moveit

```
source install/setup.bash
ros2 launch robot_moveit_config
   move_group.launch.py
```

- In a second terminal start rqt_graph:

```
source install/setup.bash
rqt_graph
ros2 node list
```

Assignment 48: Start Rviz2

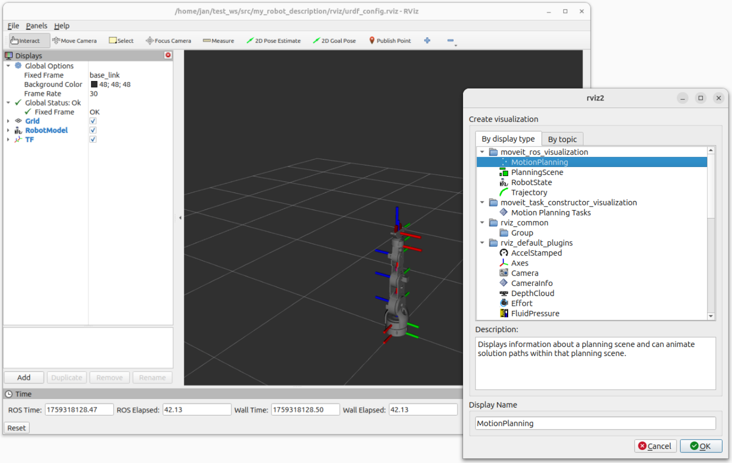
- Start rviz2

```
source install/setup.bash
ros2 run rviz2 rviz2 -d ~/minor_ws/src/
   robot_description/rviz/urdf_config.rviz
```

- In a second terminal start rqt_graph and rqt_topic:

```
rqt_graph
ros2 run rqt_topic rqt_topic
```

Assignment 48: Start Rviz2


/home/jan/test_ws/src/my_robot_description/rviz/urdf_config.rviz - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Goal Pose Publish Point
Displays
Add Duplicate Remove Rename
Time
Reset
rviz2
Create visualization
By display type By topic
Description:
Display Name
Cancel OK

Global Options
Fixed Frame base_link
Background Color 48; 48; 48
Frame Rate 30
Global Status: Ok
Fixed Frame OK
Grid ✓
RobotModel ✓
TF ✓

ROS Time: 1759318128.47 ROS Elapsed: 42.13 Wall Time: 1759318128.50 Wall Elapsed: 42.13

moveit_ros_visualization
- MotionPlanning
- PlanningScene
- RobotState
- Trajectory

moveit_task_constructor_visualization
- Motion Planning Tasks

rviz_common
- Group

rviz_default_plugins
- AccelStamped
- Axes
- Camera
- CameraInfo
- DepthCloud
- Effort
- FluidPressure

Description: Displays information about a planning scene and can animate solution paths within that planning scene.

Display Name: MotionPlanning

- Add the necessary plugins and test functionality.

Assignment 49: Build robot.launch.xml

- Now add all actions to a launch.xml file

```
<launch>
 <let name="urdf_path"
    value="$(find-pkg-share robot_description)/urdf/
       skyentific_robot.urdf.xacro" />
 <let name="rviz_config_path"
    value="$(find-pkg-share robot_bringup)/config/
       robot_moveit.rviz" />
 <node pkg="robot_state_publisher" exec="
    robot_state_publisher">
     <param name="robot_description"
        value="$(command 'xacro $(var urdf_path)')" />
 </node>
   <node pkg="controller_manager" exec="
      ros2_control_node">
     <param from="$(find-pkg-share robot_bringup)/
        config/skyentific_robot_controllers.yaml" />
 </node> ...
```

Assignment 49: Build robot.launch.xml

```xml
...
  <node pkg="controller_manager" exec="spawner" args="
      joint_state_broadcaster" />
  <node pkg="controller_manager" exec="spawner" args="
      arm_controller" />
  <node pkg="controller_manager" exec="spawner" args="
      gripper_controller" />

  <include file="$(find-pkg-share robot_moveit_config)/
      launch/move_group.launch.py" />

  <node pkg="rviz2" exec="rviz2" output="screen" args="
      -d $(var rviz_config_path)" />
</launch>
```

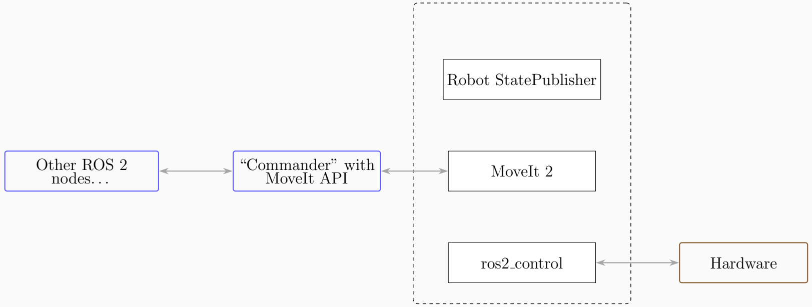
Assignment 49: Build robot.launch.xml

- start bringup package and test functionality

```
colcon build
source install/setup.bash
ros2 launch robot_bringup robot.launch.xml
```

The movit C++ API

Assignment 49: Controlling the robot



- Moveit C++ allows interaction with MoveIt without using topics, services or actions.
- Develop a reusable C++ Moveit commander template.

Assignment 50: Create packages

- Create directory structure:

```
cd ~/minor_ws/src
ros2 pkg create robot_commander --build-type
   ament_cmake --dependencies rclpp
cd my_robot_commander/src
touch test_moveit.cpp
cd ~/minor_ws/src
code .
```

- Edit CMakeLists.txt

```
code .
```

Assignment 51: edit CMakeLists.txt

```
cmake_minimum_required(VERSION 3.8)
project(robot_commander)

if(CMAKE_COMPILER_IS_GNUCXX OR CMAKE_CXX_COMPILER_ID
    MATCHES "Clang")
  add_compile_options(-Wall -Wextra -Wpedantic)
endif()

find_package(ament_cmake REQUIRED) # find dependencies
find_package(rclcpp REQUIRED)

add_executable(test_moveit src/test_moveit.cpp)
ament_target_dependencies(test_moveit rclcpp )

install(TARGETS
  test_moveit
  DESTINATION lib/${PROJECT_NAME}/
)
ament_package()
```

Assignment 52: Define test_moveit.cpp

```
#include <rclcpp/rclcpp.hpp>

int main(int argc, char **argv)
{
   rclcpp::init(argc, argv);
   auto node = std::make_shared<rclcpp::Node>("
       test_moveit");
   // we need a new thread at this point to spin node
   // - thread with instructions to move the robot
   // - thread to spin the node

   // create single threaded executor
   rclcpp::executors::SingleThreadedExecutor executor;
   executor.add_node(node);
   auto spinner = std::thread([&executor]() { executor.
       spin(); });

   // create rest of code below
```

Assignment 52: Define test_moveit.cpp

```cpp
    // create rest of code below

    rclcpp::shutdown();
    spinner.join();
    return 0;
}
```

- Start simple with straightforward C++.
- Create a main which returns 0.
- Test the build process.

```bash
colcon build && source install/setup.bash
```

Assignment 52: Define test_moveit.cpp

- Add include move_group_interface.hpp

```
#include <moveit/move_group_interface/
    move_group_interface.hpp>
```

- Save to activate autocomplete (edit c_cpp_properties.json).
- Then add after "create rest of code below"

```
  auto arm = moveit::planning_interface::
      MoveGroupInterface(node, "arm");
  arm.setMaxVelocityScalingFactor(1.0);
  arm.setMaxAccelerationScalingFactor(1.0);

  auto gripper = moveit::planning_interface::
      MoveGroupInterface(node, "gripper");
  // Add some moves from named goals from robot.srdf
```

Assignment 53: Edit skyentific_robot.srdf

```
...
<group_state name="pose_1" group="arm">
      <joint name="joint_1" value="-0.5235"/>
      <joint name="joint_2" value="-0.786"/>
      <joint name="joint_3" value="-0.786"/>
      <joint name="joint_4" value="-1.571"/>
      <joint name="joint_5" value="-0.5235"/>
   </group_state>
   <group_state name="pose_2" group="arm">
      <joint name="joint_1" value="0.5235"/>
      <joint name="joint_2" value="-0.786"/>
      <joint name="joint_3" value="-0.786"/>
      <joint name="joint_4" value="-1.571"/>
      <joint name="joint_5" value="0.5235"/>
   </group_state>
...
```

Assignment 53: Define test_moveit.cpp

- Set start state to current state and add target

```
arm.setStartStateToCurrentState();
arm.setNamedTarget("pose_1");
```

- Then add a plan and execute if planning is succesfull

```
moveit::planning_interface::MoveGroupInterface::Plan
    plan1;
bool success1 = (arm.plan(plan1) == moveit::core::
    MoveItErrorCode::SUCCESS);

if (success1) {
   arm.execute(plan1);
}
```

- build and test package

Assignment 54: edit CMakeLists.txt

- We need to add a package to CMakeLists.txt

```
...
find_package(ament_cmake REQUIRED) # find dependencies
find_package(rclcpp REQUIRED)
find_package(moveit_ros_planning_interface REQUIRED)

add_executable(test_moveit src/test_moveit.cpp)
ament_target_dependencies(test_moveit rclcpp
    moveit_ros_planning_interface)

install(TARGETS
  test_moveit
  DESTINATION lib/${PROJECT_NAME}/
)
ament_package()
```

Assignment 55: Edit package.xml and execute

- We need to edit package.xml

```
...
 <depend>rclpp</depend>
 <depend>moveit_ros_planning_interface</depend>
...
```

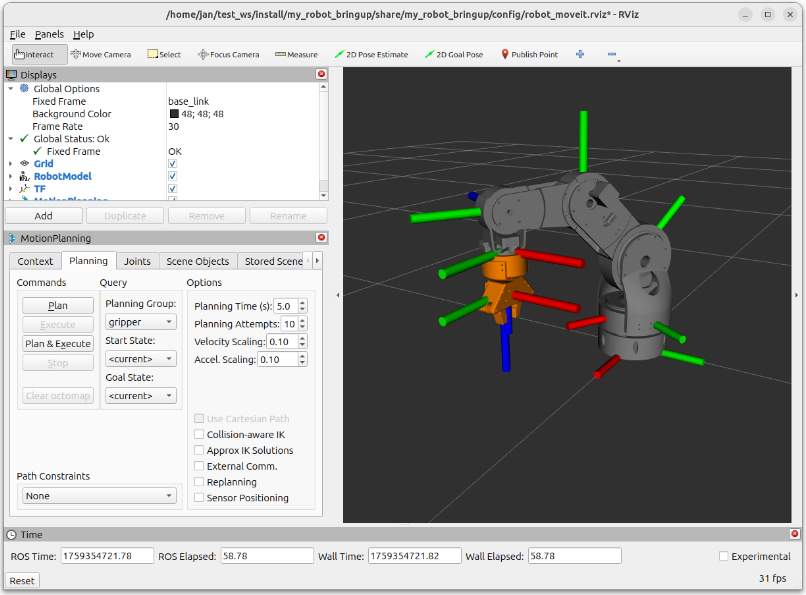
- Then build and execute:

```
colcon build && source install/setup.bash
ros2 launch robot_bringup robot.launch.xml
```

- In a second terminal run the commander

```
source install/setup.bash
ros2 run robot_commander test_moveit
```

Assignment 55: Test controlling the robot



/home/jan/test_ws/install/my_robot_bringup/share/my_robot_bringup/config/robot_moveit.rviz* - RViz

Global Options
Fixed Frame: base_link
Background Color: 48; 48; 48
Frame Rate: 30
Global Status: Ok
Fixed Frame: OK
Grid ✓
RobotModel ✓
TF ✓

Planning Group: gripper
Start State: <current>
Goal State: <current>
Path Constraints: None

Planning Time (s): 5.0
Planning Attempts: 10
Velocity Scaling: 0.10
Accel. Scaling: 0.10

Use Cartesian Path ☐
Collision-aware IK ☐
Approx IK Solutions ☐
External Comm. ☐
Replanning ☐
Sensor Positioning ☐

Time
ROS Time: 1759354721.78 ROS Elapsed: 58.78 Wall Time: 1759354721.82 Wall Elapsed: 58.78 ☐ Experimental

31 fps