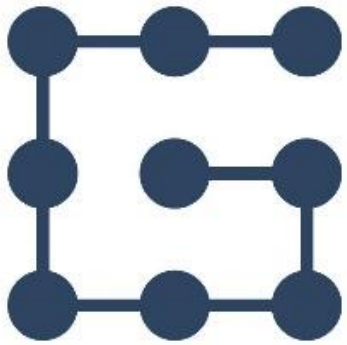


ROS机械臂开发

—— 6.机械臂仿真系统

目录

目录

1.完善机器人模型中的物理属性

第一步：为link添加惯性参数和碰撞属性

```
<!-- Macro for inertia matrix -->
<xacro:macro name="cylinder_inertial_matrix" params="m r h">
    <inertial>
        <mass value="${m}" />
        <inertia ixx="${m*(3*r*r+h*h)/12}" ixy = "0" ixz = "0"
            iyy="${m*(3*r*r+h*h)/12}" iyz = "0"
            izz="${m*r*r/2}" />
    </inertial>
</xacro:macro>

<!-- ////////////////////////////////////    LINK0    ////////////////////////////////////////////// -->
<link name="link0">
    <visual>
        <origin xyz="0 0 0" rpy="0 0 0" />
        <geometry>
            <cylinder radius="${link0_radius}" length="${link0_length}"/>
        </geometry>
        <material name="White" />
    </visual>
    <collision>
        <origin xyz="0 0 0" rpy="0 0 0" />
        <geometry>
            <cylinder radius="${link0_radius}" length="${link0_length}"/>
        </geometry>
    </collision>
    <cylinder_inertial_matrix m="${link0_mass}" r="${link0_radius}" h="${link0_length}"/>
</link>
```

1.完善机器人模型中的物理属性

第二步：为link添加gazebo标签

```
<!-- ////////////////////////////////////   Gazebo   //////////////////////////////////////// -->
<gazebo reference="bottom_link">
    <material>Gazebo/White</material>
</gazebo>
<gazebo reference="base_link">
    <material>Gazebo/White</material>
</gazebo>
<gazebo reference="link1">
    <material>Gazebo/Blue</material>
</gazebo>
<gazebo reference="link2">
    <material>Gazebo/White</material>
</gazebo>
<gazebo reference="link3">
    <material>Gazebo/Blue</material>
</gazebo>
<gazebo reference="link4">
    <material>Gazebo/Black</material>
</gazebo>
<gazebo reference="link5">
    <material>Gazebo/White</material>
</gazebo>
<gazebo reference="link6">
    <material>Gazebo/Blue</material>
</gazebo>
<gazebo reference="gripper_finger_link1">
    <material>Gazebo/White</material>
</gazebo>
<gazebo reference="gripper_finger_link2">
    <material>Gazebo/White</material>
</gazebo>
```

1.完善机器人模型中的物理属性

第三步：为joint添加传动装置

```
<!-- Transmissions for ROS Control -->
<xacro:macro name="transmission_block" params="joint_name">
    <transmission name="${joint_name}_trans">
        <type>transmission_interface/SimpleTransmission</type>
        <joint name="${joint_name}">
            <hardwareInterface>hardware_interface/PositionJointInterface</hardwareInterface>
        </joint>
        <actuator name="${joint_name}_motor">
            <hardwareInterface>hardware_interface/PositionJointInterface</hardwareInterface>
            <mechanicalReduction>1</mechanicalReduction>
        </actuator>
    </transmission>
</xacro:macro>

<xacro:transmission_block joint_name="joint1"/>
<xacro:transmission_block joint_name="joint2"/>
<xacro:transmission_block joint_name="joint3"/>
<xacro:transmission_block joint_name="joint4"/>
<xacro:transmission_block joint_name="joint5"/>
<xacro:transmission_block joint_name="joint6"/>
<xacro:transmission_block joint_name="finger_joint1"/>
```

1.完善机器人模型中的物理属性

第四步：添加gazebo控制器插件

```
<!-- ros_control plugin -->
<gazebo>
    <plugin name="gazebo_ros_control" filename="libgazebo_ros_control.so">
        <robotNamespace>/marm</robotNamespace>
    </plugin>
</gazebo>
```

<robotNamespace>：机器人的命名空间

1.完善机器人模型中的物理属性

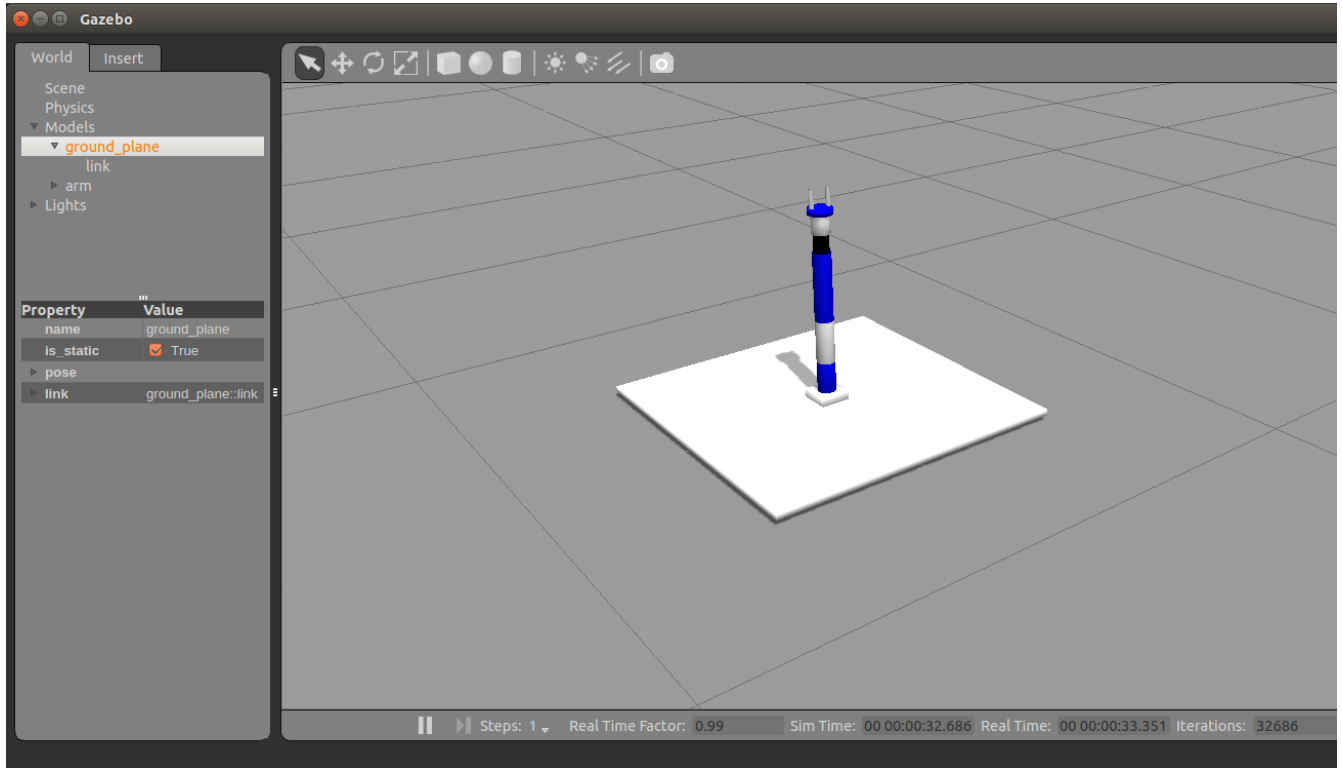
在gazebo中加载机器人模型

```
<launch>

  <!-- these are the arguments you can pass this launch file, for example paused:=true -->
  <arg name="paused" default="false"/>
  <arg name="use_sim_time" default="true"/>
  <arg name="gui" default="true"/>
  <arg name="headless" default="false"/>
  <arg name="debug" default="false"/>

  <!-- We resume the logic in empty_world.launch -->
  <include file="$(find gazebo_ros)/launch/empty_world.launch">
    <arg name="debug" value="$(arg debug)" />
    <arg name="gui" value="$(arg gui)" />
    <arg name="paused" value="$(arg paused)"/>
    <arg name="use_sim_time" value="$(arg use_sim_time)"/>
    <arg name="headless" value="$(arg headless)"/>
  </include>

  <!-- Load the URDF into the ROS Parameter Server -->
  <param name="robot_description" command="$(find xacro)/xacro --inorder '$(find marm_description)/urdf/marm.xacro'" />

  <!-- Run a python script to the send a service call to gazebo_ros to spawn a URDF robot -->
  <node name="urdf_spawner" pkg="gazebo_ros" type="spawn_model" respawn="false" output="screen"
    args="-urdf -model marm -param robot_description"/>

</launch>
```

marm_gazebo/launch/marm_world.launch

1.完善机器人模型中的物理属性

启动仿真环境　　$ roslaunch marm_gazebo marm_world.launch

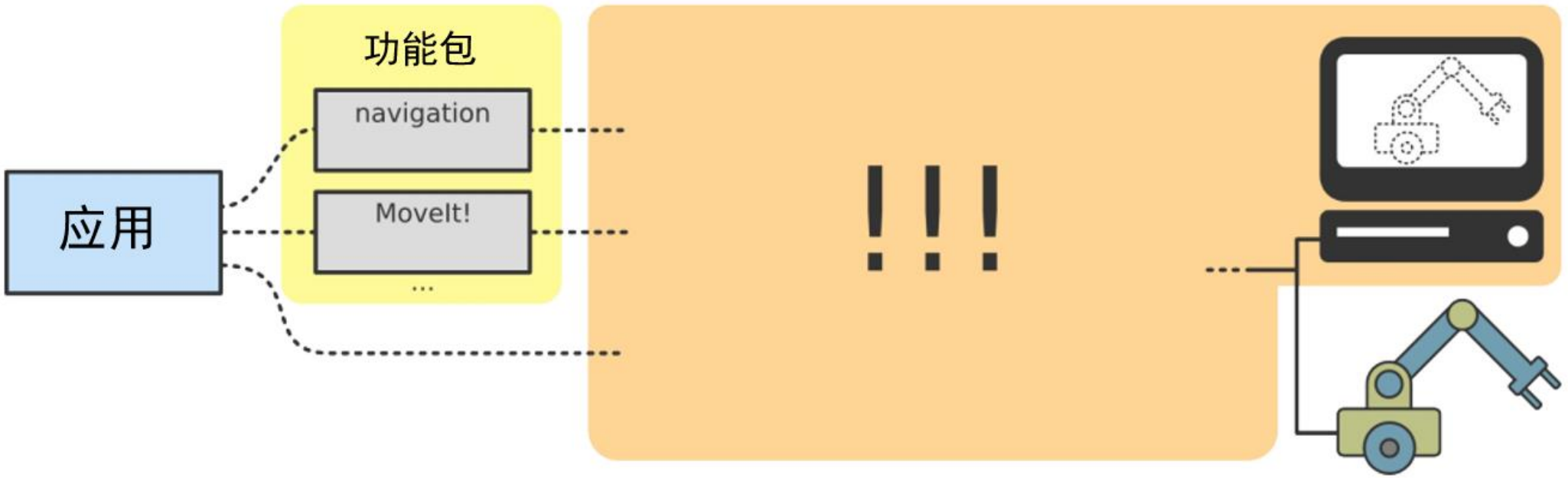


建议：为保证模型顺利加载，请提前将模型文件库下载并放置到~/.gazebo/models下

https://bitbucket.org/osrf/gazebo_models/downloads/

目录

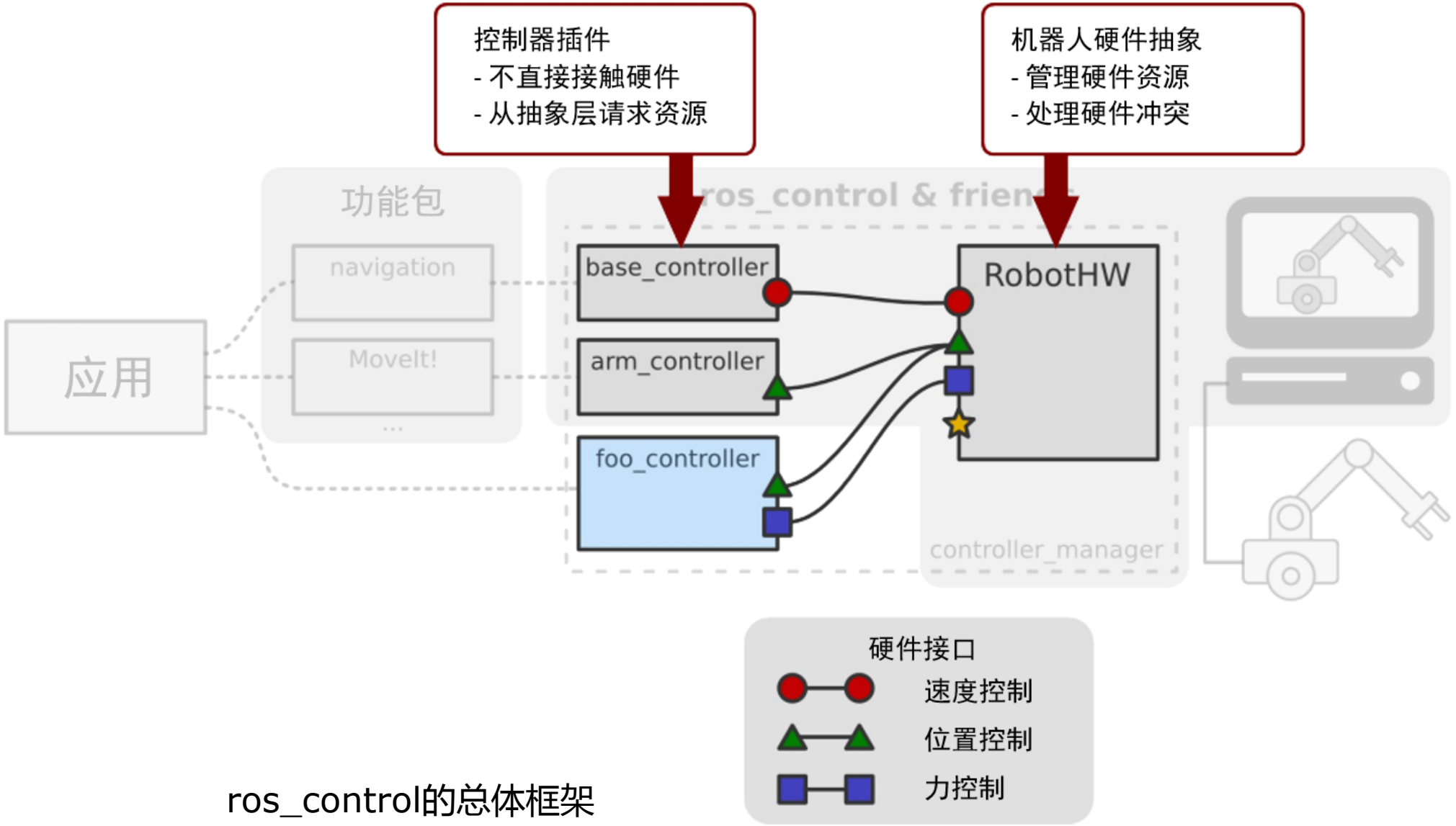
2. ROS中的控制器ros_control



ros_control是什么？

- ROS为开发者提供的机器人控制中间件
- 包含一系列控制器接口、传动装置接口、硬件接口、控制器工具箱等等
- 可以帮助机器人应用功能包快速落地，提高开发效率

2. ROS中的控制器ros_control



ros_control的总体框架

2. ROS中的控制器ros_control

- 控制器管理器

提供一种通用的接口来管理不同的控制器。

- 控制器

读取硬件状态，发布控制命令，完成每个joint的控制。

- 硬件资源

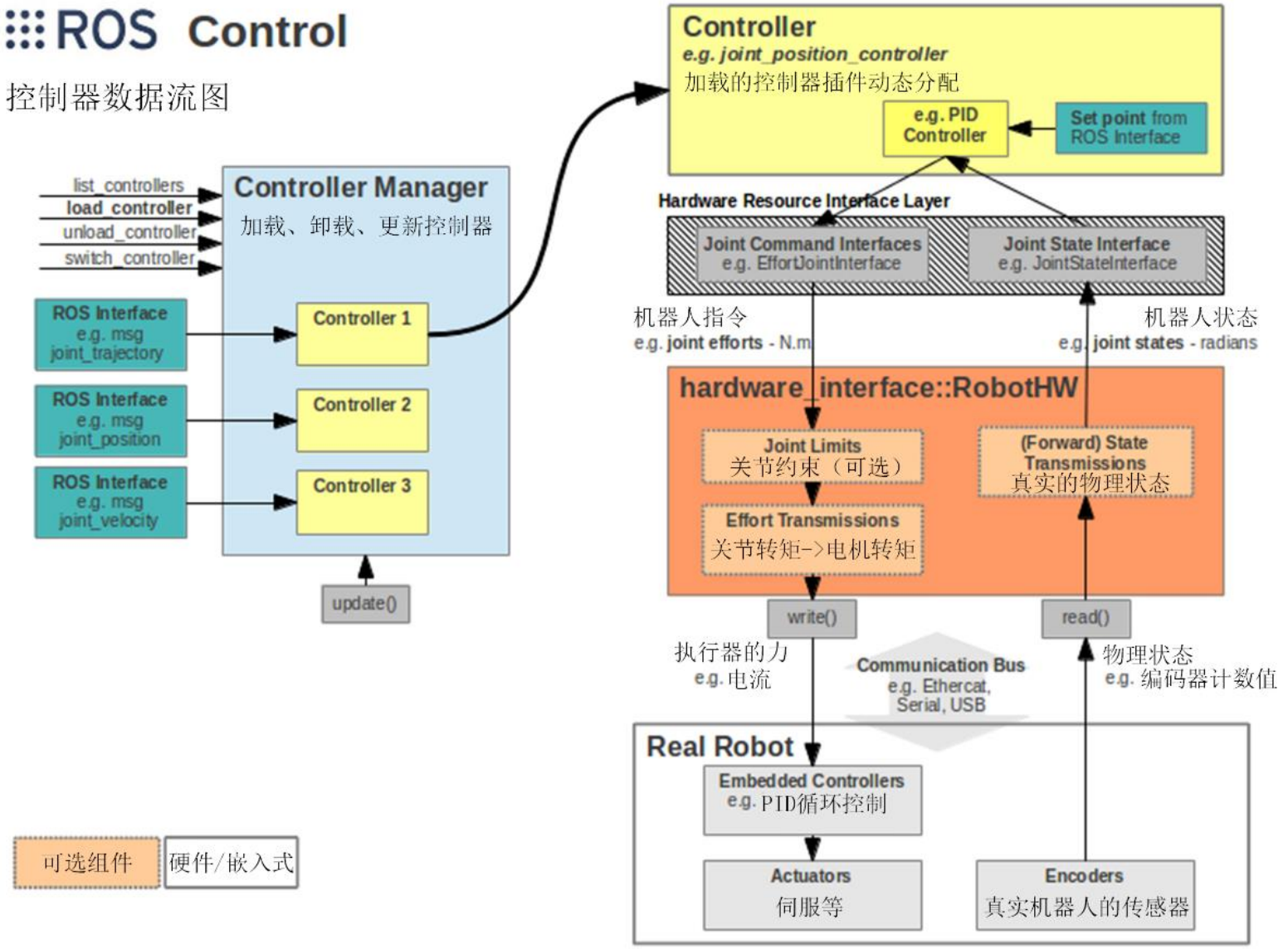
为上下两层提供硬件资源的接口。

- 机器人硬件抽象

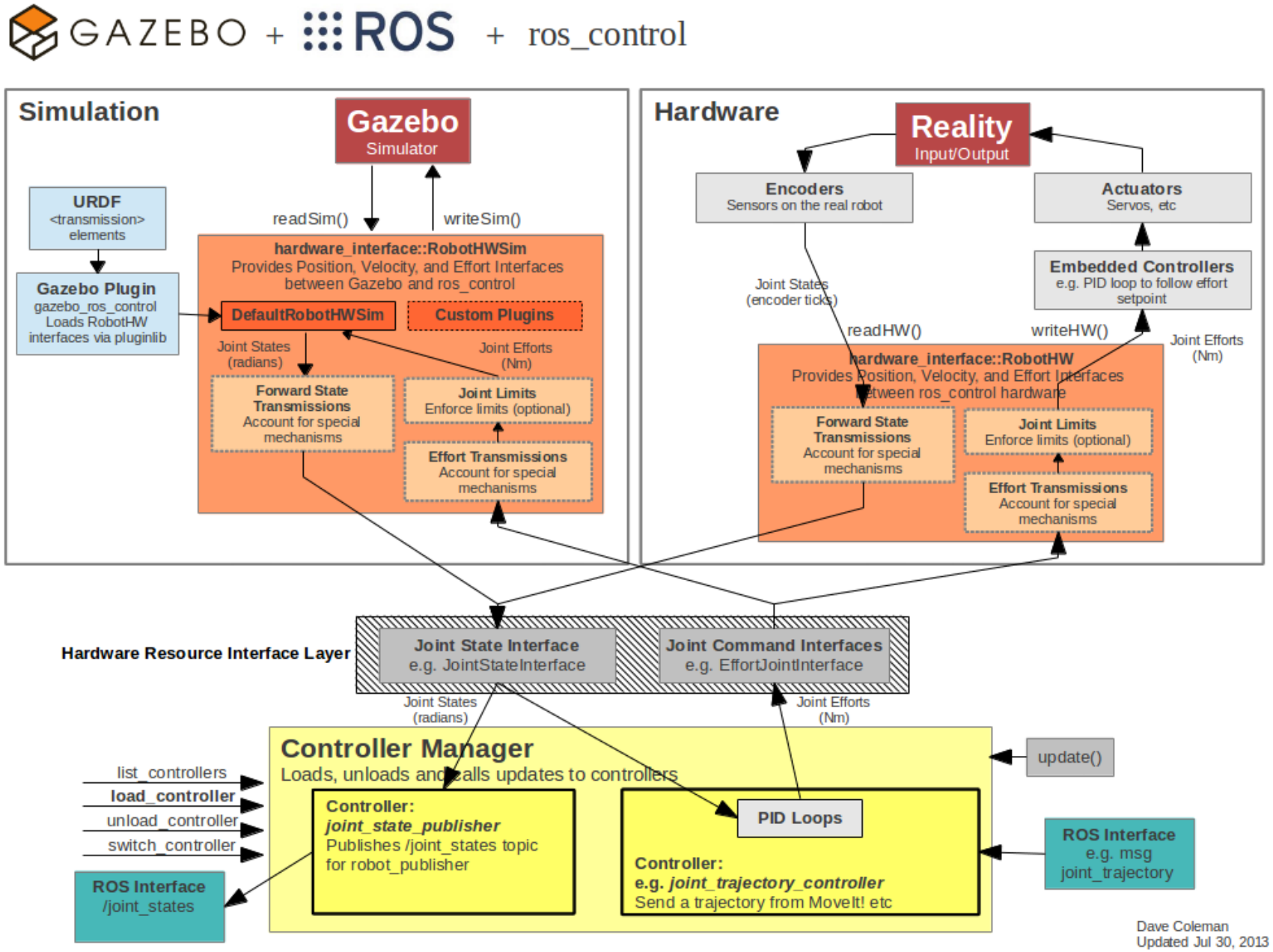
机器人硬件抽象和硬件资源直接打交道，通过write和read方法完成硬件操作。

- 真实机器人

执行接收到的命令。



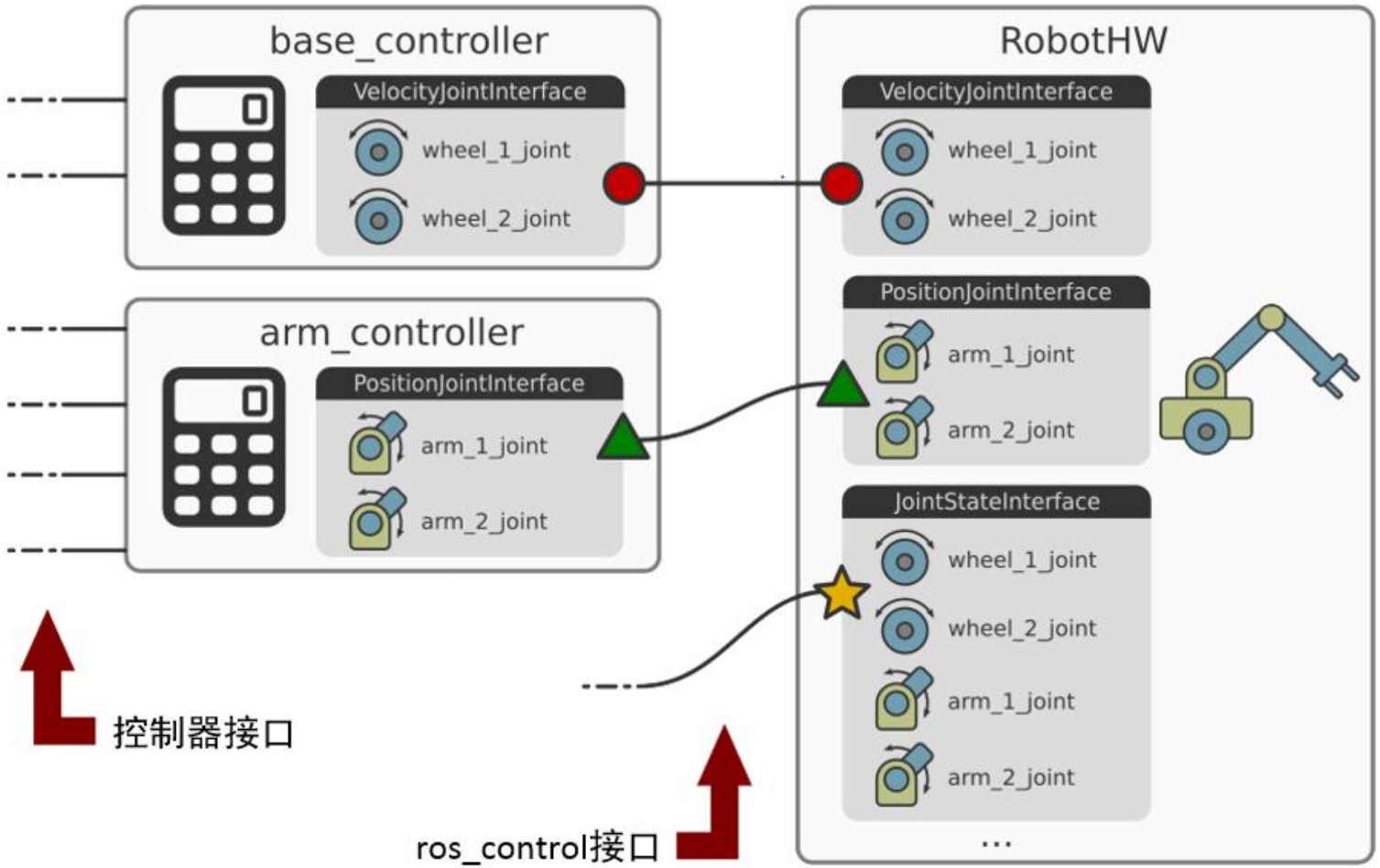
2. ROS中的控制器ros_control



2. ROS中的控制器ros_control

控制器（Controllers）：

- joint_state_controller
- joint_effort_controller
- joint_position_controller
- joint_velocity_controller



可参考：https://github.com/ros-controls/ros_control/wiki/controller_interface

目录

3. Gazebo构建机械臂仿真系统 —— 配置控制器参数

关节位置控制器

Joint Position Controller

marm_gazebo/config/marm_gazebo_control.yaml

```yaml
marm:
  # Publish all joint states -----------------------------
  joint_state_controller:
    type: joint_state_controller/JointStateController
    publish_rate: 50

  # Position Controllers ---------------------------------
  joint1_position_controller:
    type: position_controllers/JointPositionController
    joint: joint1
    pid: {p: 100.0, i: 0.01, d: 10.0}
  joint2_position_controller:
    type: position_controllers/JointPositionController
    joint: joint2
    pid: {p: 100.0, i: 0.01, d: 10.0}
  joint3_position_controller:
    type: position_controllers/JointPositionController
    joint: joint3
    pid: {p: 100.0, i: 0.01, d: 10.0}
  joint4_position_controller:
    type: position_controllers/JointPositionController
    joint: joint4
    pid: {p: 100.0, i: 0.01, d: 10.0}
  joint5_position_controller:
    type: position_controllers/JointPositionController
    joint: joint5
    pid: {p: 100.0, i: 0.01, d: 10.0}
  joint6_position_controller:
    type: position_controllers/JointPositionController
    joint: joint6
    pid: {p: 100.0, i: 0.01, d: 10.0}
```

3. Gazebo构建机械臂仿真系统 —— 加载控制器参数

```
<launch>

    <!-- 将关节控制器的配置参数加载到参数服务器中 -->
    <rosparam file="$(find marm_gazebo)/config/marm_gazebo_control.yaml" command="load"/>

    <!-- 加载controllers -->
    <node name="controller_spawner" pkg="controller_manager" type="spawner" respawn="false"
          output="screen" ns="/marm" args="joint_state_controller
                                           joint1_position_controller
                                           joint2_position_controller
                                           joint3_position_controller
                                           joint4_position_controller
                                           joint5_position_controller
                                           joint6_position_controller"/>

    <!-- 运行robot_state_publisher节点，发布tf  -->
    <node name="robot_state_publisher" pkg="robot_state_publisher" type="robot_state_publisher"
          respawn="false" output="screen">
        <remap from="/joint_states" to="/marm/joint_states" />
    </node>

</launch>
```

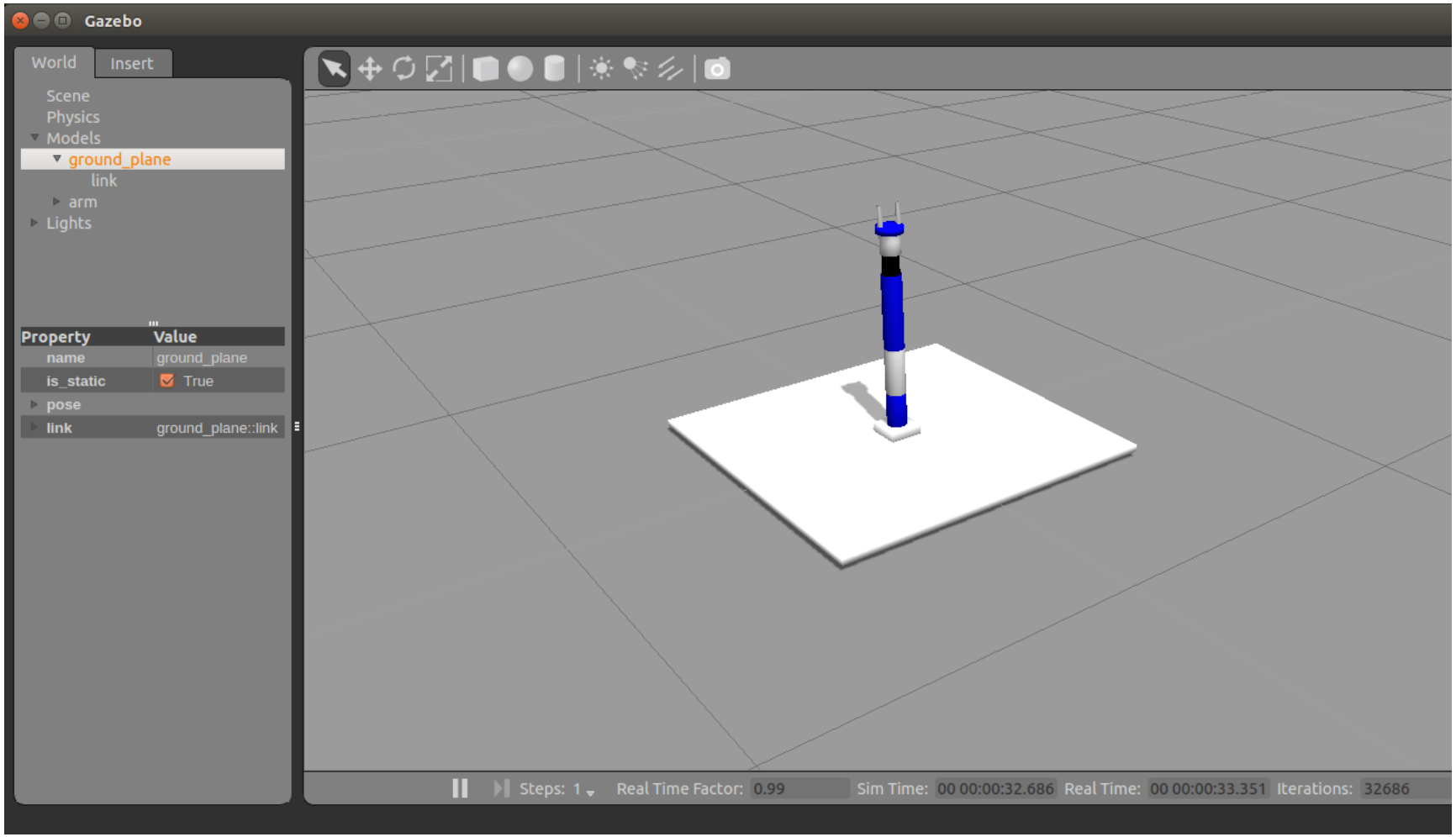
marm_gazebo/launch/marm_gazebo_controller.launch

3. Gazebo构建机械臂仿真系统 —— 加载控制器参数

```
<launch>

    <!-- 启动Gazebo  -->
    <include file="$(find marm_gazebo)/launch/marm_world.launch" />

    <!-- 启动Gazebo controllers -->
    <include file="$(find marm_gazebo)/launch/marm_gazebo_controller.launch" />

</launch>
```

marm_gazebo/launch/marm_gazebo_control.launch

3. Gazebo构建机械臂仿真系统 —— 启动仿真



启动仿真系统　$ roslaunch marm_gazebo marm_gazebo_control.launch

3. Gazebo构建机械臂仿真系统 —— 启动仿真

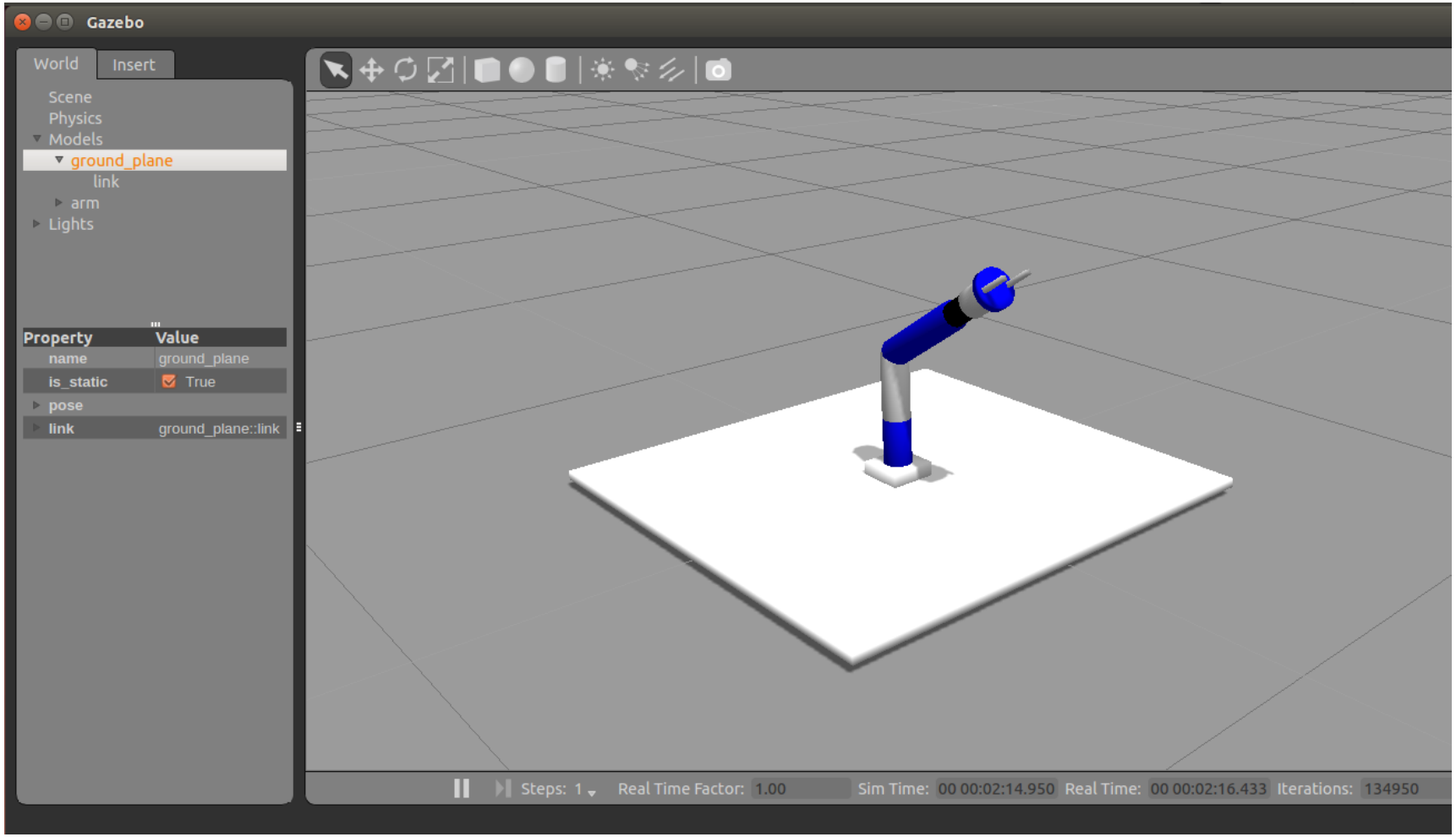
```
[ INFO] [1489676219.749984313, 0.133000000]: Loaded gazebo_ros_control.
[INFO] [WallTime: 1489676220.002000] [0.358000] Controller Spawner: Waiting for service controller_manager/switch_controller
[INFO] [WallTime: 1489676220.005710] [0.359000] Controller Spawner: Waiting for service controller_manager/unload_controller
[INFO] [WallTime: 1489676220.009109] [0.362000] Loading controller: joint_state_controller
[INFO] [WallTime: 1489676220.075714] [0.385000] Loading controller: joint1_position_controller
[INFO] [WallTime: 1489676220.139417] [0.434000] Loading controller: joint2_position_controller
[INFO] [WallTime: 1489676220.156092] [0.453000] Loading controller: joint3_position_controller
[INFO] [WallTime: 1489676220.187286] [0.485000] Loading controller: joint4_position_controller
[INFO] [WallTime: 1489676220.204836] [0.498000] Loading controller: joint5_position_controller
```

制器插件加载成功的日志信息

```
/clock
/gazebo/link_states
/gazebo/model_states
/gazebo/parameter_descriptions
/gazebo/parameter_updates
/gazebo/set_link_state
/gazebo/set_model_state
/marm/joint1_position_controller/command
/marm/joint2_position_controller/command
/marm/joint3_position_controller/command
/marm/joint4_position_controller/command
/marm/joint5_position_controller/command
/marm/joint6_position_controller/command
/marm/joint_states
```

查看controller订阅的控制命令话题

3. Gazebo构建机械臂仿真系统 —— 启动仿真



控制单轴运动 $ rostopic pub /marm/joint2_position_controller/command std_msgs/Float64 1.0

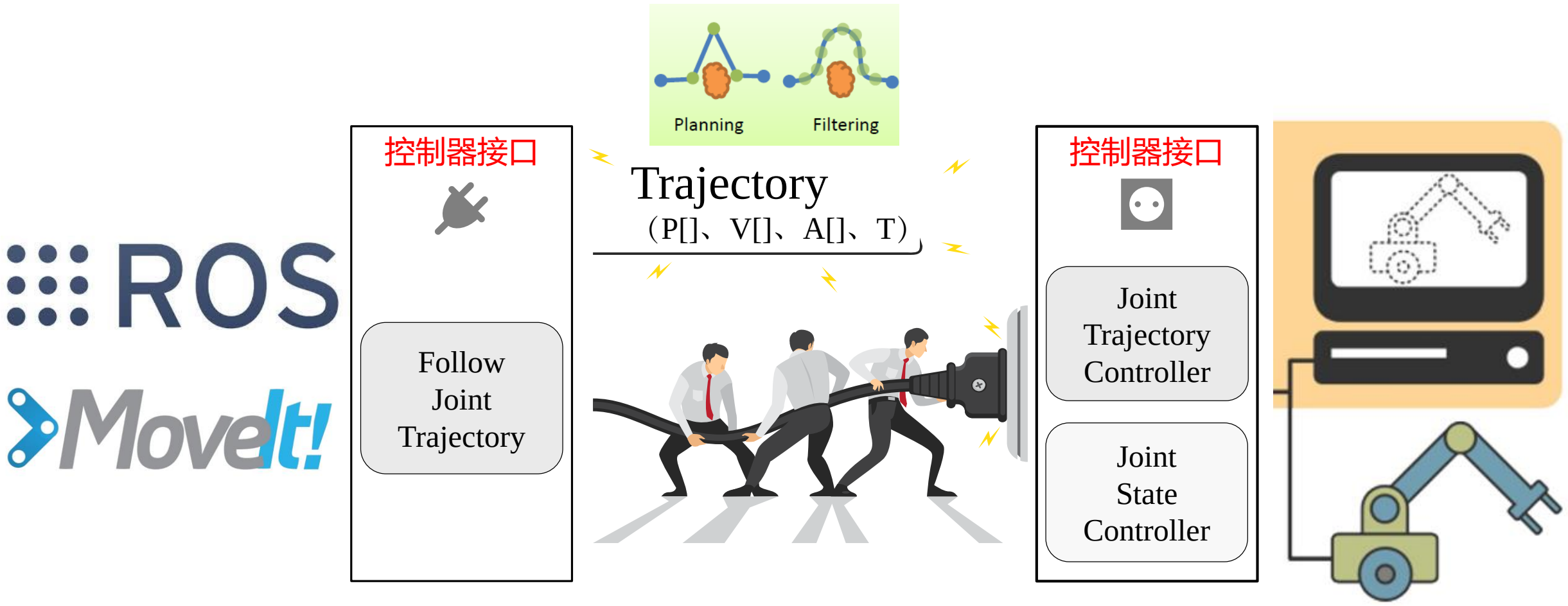
3. Gazebo构建机械臂仿真系统 —— 启动仿真

```
➜  ~ rostopic echo /marm/joint_states
header:
  seq: 2705
  stamp:
    secs: 54
    nsecs: 374000000
  frame_id: ''
name: [finger_joint1, joint1, joint2, joint3, joint4, joint5, joint6]
position: [-2.297278393051281e-09, 2.0262235112866733e-05, 1.0000067750528645, 1.0828529998008207e-05, -1.1676260514015269e-05, -2
.2574341514314256e-05, 4.0627083933486574e-05]
velocity: [3.268659395106492e-06, 8.869627266851535e-05, 0.006775099434874977, 0.011243363098868208, 7.571551171683708e-05, -0.023
059533324612046, 0.00010223269118469 58]
effort: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
---
header:
  seq: 2706
  stamp:
    secs: 54
    nsecs: 394000000
  frame_id: ''
name: [finger_joint1, joint1, joint2, joint3, joint4, joint5, joint6]
position: [-4.707067854674011e-09, 2.026646390707043e-05, 1.0000067416898961, 1.0864115932207596e-05, -1.1679788002538771e-05, -2.
2443820867579234e-05, 4.0591541702994505e-05]
velocity: [-1.0844295819972196e-05, 7.348242827360026e-05, 0.00674173584213059, 0.011355066799606626, 9.153942025746042e-05, -0.02
2957056213183596, 0.00013044254351341405]
effort: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
---
```

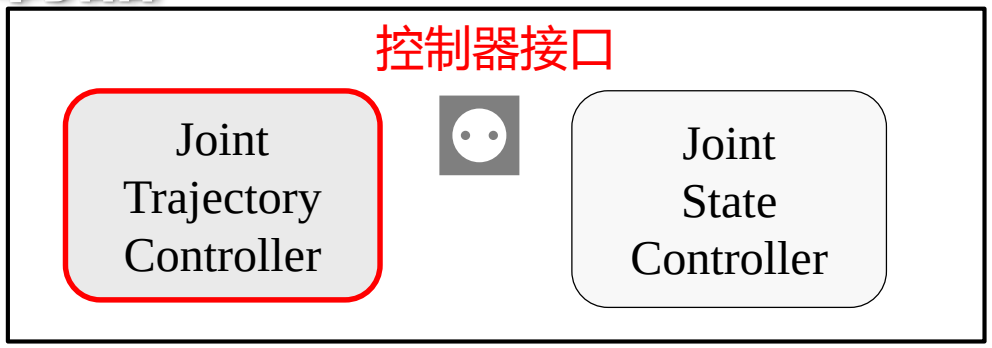
监控机器人状态　　$ rostopic echo /marm/joint_states

目录

4. MoveIt!+Gazebo构建机械臂仿真系统



4. MoveIt!+Gazebo构建机械臂仿真系统 —— 配置控制器



关节轨迹控制器

- 线性样条：位置连续，速度、加速度不连续。
- 三次样条：位置和速度连续，加速度不连续。
- 五次样条：位置、速度、加速度都连续。

```yaml
marm:
  arm_joint_controller:
    type: "position_controllers/JointTrajectoryController"
    joints:
      - joint1
      - joint2
      - joint3
      - joint4
      - joint5
      - joint6

    gains:
      joint1:    {p: 1000.0, i: 0.0, d: 0.1, i_clamp: 0.0}
      joint2:    {p: 1000.0, i: 0.0, d: 0.1, i_clamp: 0.0}
      joint3:    {p: 1000.0, i: 0.0, d: 0.1, i_clamp: 0.0}
      joint4:    {p: 1000.0, i: 0.0, d: 0.1, i_clamp: 0.0}
      joint5:    {p: 1000.0, i: 0.0, d: 0.1, i_clamp: 0.0}
      joint6:    {p: 1000.0, i: 0.0, d: 0.1, i_clamp: 0.0}


  gripper_controller:
    type: "position_controllers/JointTrajectoryController"
    joints:
      - finger_joint1
    gains:
      finger_joint1:  {p: 50.0, d: 1.0, i: 0.01, i_clamp: 1.0}
```

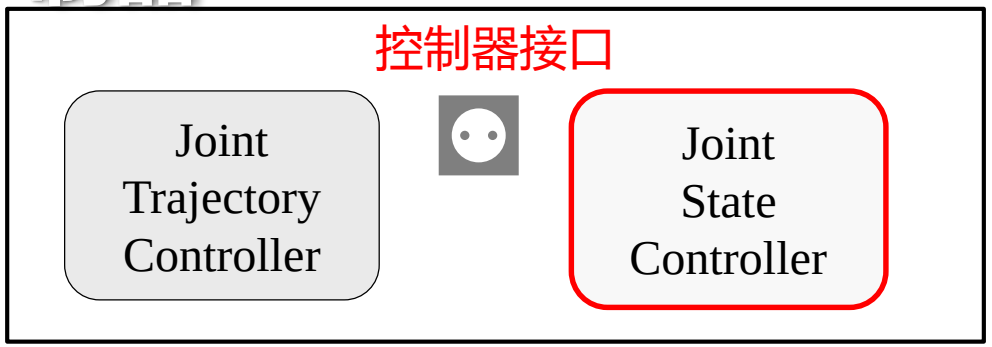
marm_gazebo/config/trajectory_control.yaml

4. MoveIt!+Gazebo构建机械臂仿真系统 —— 加载控制器

```
<launch>

    <rosparam file="$(find marm_gazebo)/config/trajectory_control.yaml" command="load"/>

    <node name="arm_controller_spawner" pkg="controller_manager" type="spawner" respawn="false"
          output="screen" ns="/marm" args="arm_joint_controller gripper_controller"/>

</launch>
```

marm_gazebo/launch/marm_trajectory_controller.launch

4. MoveIt!+Gazebo构建机械臂仿真系统 —— 配置控制器




```yaml
marm:
  # Publish all joint states -----------------------------
  joint_state_controller:
    type: joint_state_controller/JointStateController
    publish_rate: 50
```

marm_gazebo/config/marm_gazebo_joint_states.yaml

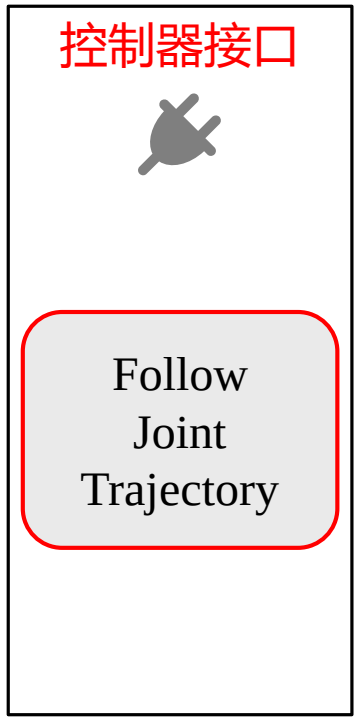
关节状态控制器

```xml
<launch>
    <!-- 将关节控制器的配置参数加载到参数服务器中 -->
    <rosparam file="$(find marm_gazebo)/config/marm_gazebo_joint_states.yaml" command="load"/>

    <node name="joint_controller_spawner" pkg="controller_manager" type="spawner" respawn="false"
          output="screen" ns="/marm" args="joint_state_controller" />

    <!-- 运行robot_state_publisher节点，发布tf  -->
    <node name="robot_state_publisher" pkg="robot_state_publisher" type="robot_state_publisher"
        respawn="false" output="screen">
        <remap from="/joint_states" to="/marm/joint_states" />
    </node>

</launch>
```

marm_gazebo/launch/marm_gazebo_states.launch

4. MoveIt!+Gazebo构建机械臂仿真系统 —— 配置控制器

MoveIt!控制器




```yaml
controller_manager_ns: controller_manager
controller_list:
  - name: marm/arm_joint_controller
    action_ns: follow_joint_trajectory
    type: FollowJointTrajectory
    default: true
    joints:
      - joint1
      - joint2
      - joint3
      - joint4
      - joint5
      - joint6

  - name: marm/gripper_controller
    action_ns: follow_joint_trajectory
    type: FollowJointTrajectory
    default: true
    joints:
      - finger_joint1
      - finger_joint2
```

marm_moveit_config/config/controllers.yaml

4. MoveIt!+Gazebo构建机械臂仿真系统 —— 配置控制器

```
<launch>
  <!-- Set the param that trajectory_execution_manager needs to find the controller plugin -->
  <arg name="moveit_controller_manager" default="moveit_simple_controller_manager/MoveItSimpleControllerManager" />
  <param name="moveit_controller_manager" value="$(arg moveit_controller_manager)"/>

  <!-- load controller_list -->
  <!-- Gazebo -->
  <rosparam file="$(find marm_moveit_config)/config/controllers.yaml"/>
</launch>
```

marm_moveit_config/launch/marm_moveit_controller_manager.launch

```
[ INFO] [1505707140.146328172, 10.268000000]: Waiting for arm_controller/follow_joint_
trajectory to come up
[ERROR] [1505707145.167116253, 15.268000000]: Action client not connected: arm_control
ler/follow_joint_trajectory
```

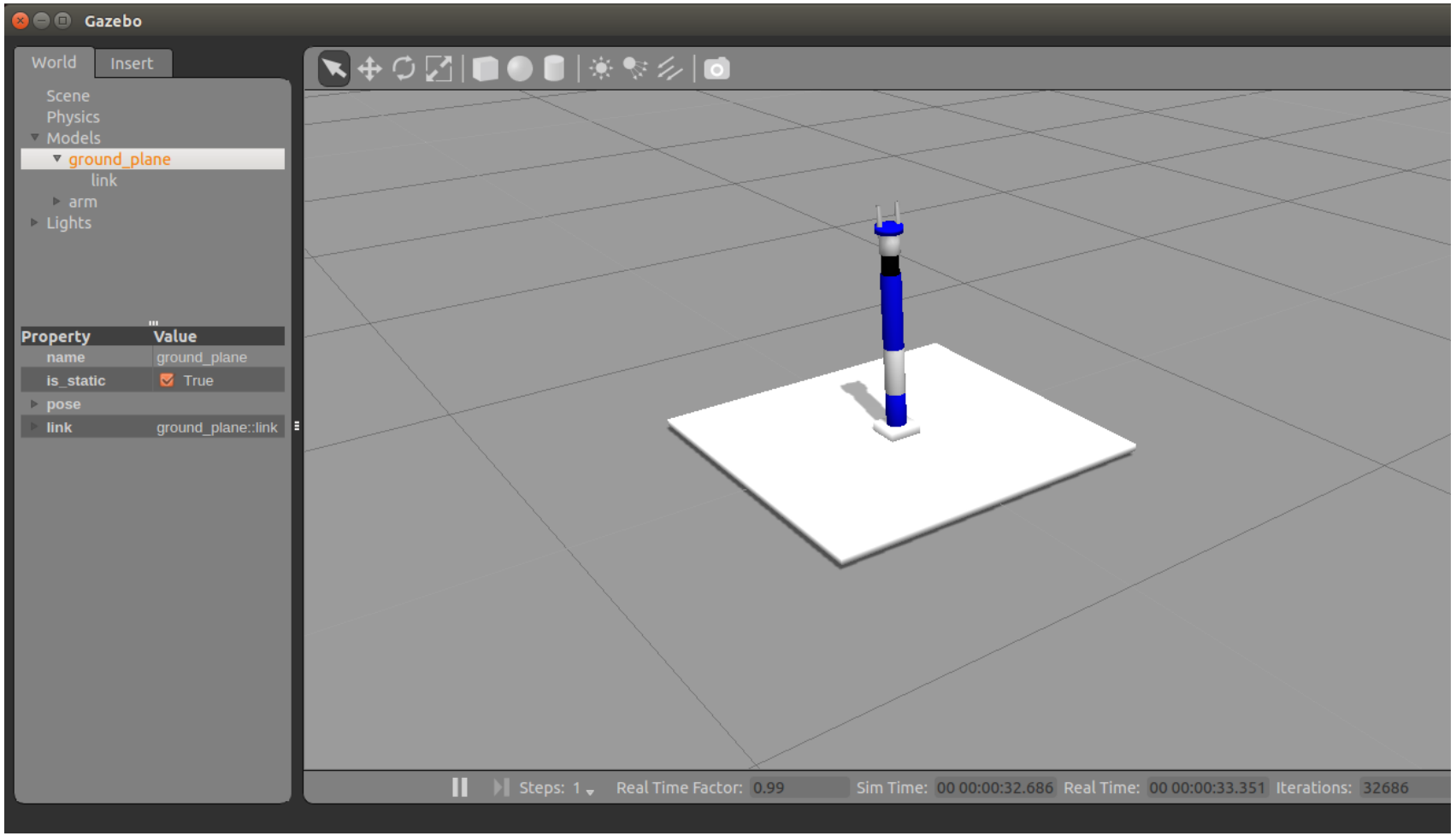
controller配置错误导致的错误

4. MoveIt!+Gazebo构建机械臂仿真系统 —— 启动仿真系统

```
<launch>

    <!-- Launch Gazebo  -->
    <include file="$(find marm_gazebo)/launch/marm_world.launch" />

    <!-- ros_control arm launch file -->
    <include file="$(find marm_gazebo)/launch/marm_gazebo_states.launch" />

    <!-- ros_control trajectory control dof arm launch file -->
    <include file="$(find marm_gazebo)/launch/marm_trajectory_controller.launch" />

    <!-- moveit launch file -->
    <include file="$(find marm_moveit_config)/launch/moveit_planning_execution.launch" />

</launch>
```

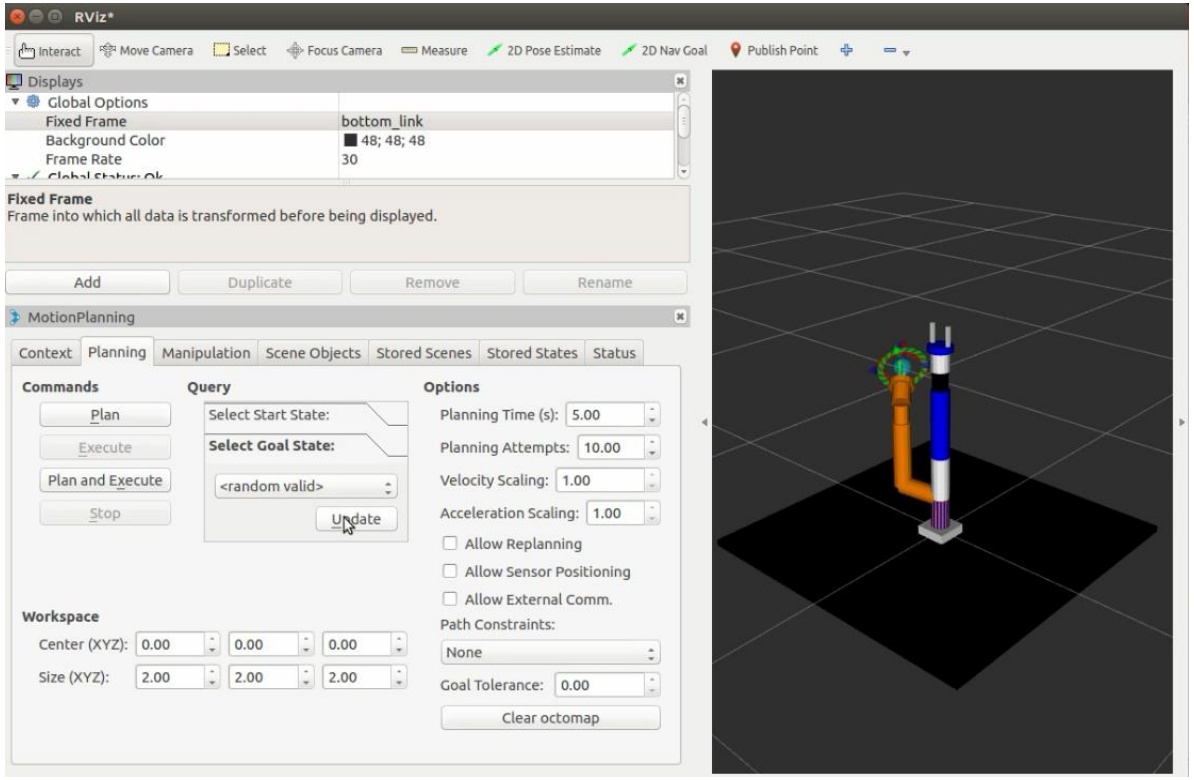
marm_gazebo/launch/marm_bringup_moveit.launch

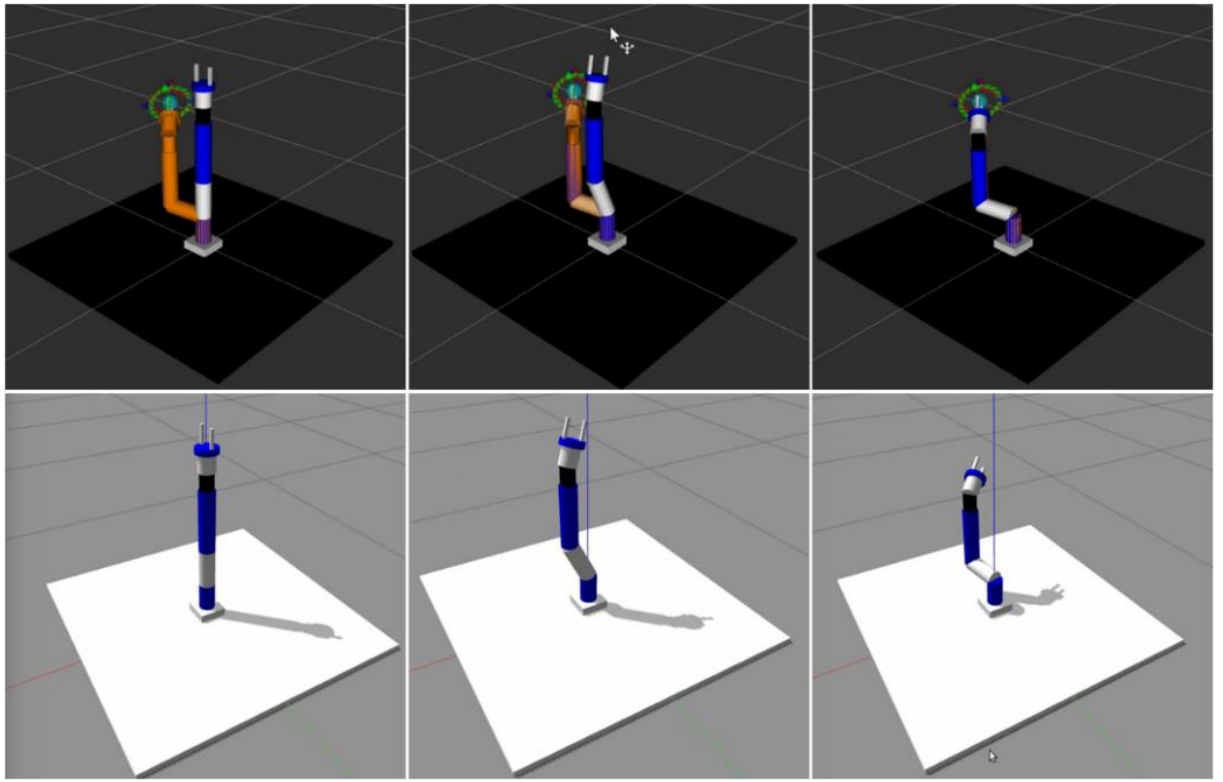
4. MoveIt!+Gazebo构建机械臂仿真系统 —— 启动仿真系统



启动仿真系统　$ roslaunch marm_gazebo marm_bringup_moveit.launch

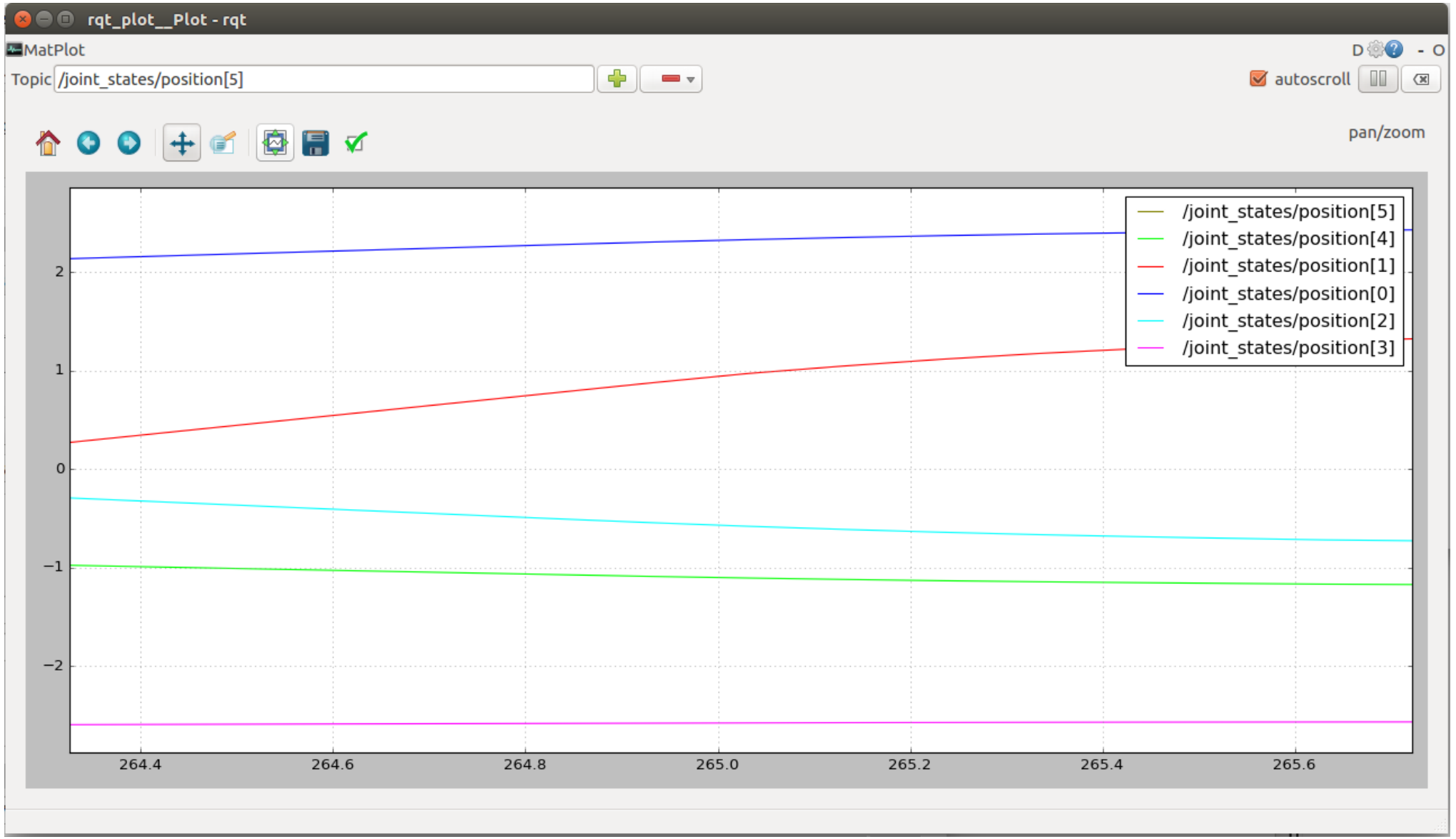
4. MoveIt!+Gazebo构建机械臂仿真系统 —— 控制仿真





通过MoveIt!控制机械臂运动，gazebo仿真环境和rviz中的机器人状态保持一致

4. MoveIt!+Gazebo构建机械臂仿真系统 —— 控制仿真



可视化显示各轴的运动状态

4. MoveIt!+Gazebo构建机械臂仿真系统 —— 控制仿真

Trajectory

```
→  ~ rostopic echo /marm/arm_joint_controller/follow_joint_trajectory/goal
WARNING: no messages received and simulated time is active.
Is /clock being published?
header:
  seq: 4
  stamp:
    secs: 146
    nsecs: 201000000
  frame_id: ''
goal_id:
  stamp:
    secs: 146
    nsecs: 201000000
  id: "/move_group-5-146.201000000"
goal:
  trajectory:
    header:
      seq: 0
      stamp:
        secs: 0
        nsecs:         0
      frame_id: "/bottom_link"
    joint_names: [joint1, joint2, joint3, joint4, joint5, joint6]
    points:
      -
        positions: [2.107812371248425, 0.22625739904110986, 1.2865145106142792, -0.14124708751311577, 0.2784896219480171, 2.8361966249022883]
        velocities: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
        accelerations: [-1.0001071448355183, 0.0, 0.0, 0.0, 0.0, 0.0]
        effort: []
        time_from_start:
          secs: 0
          nsecs:         0
      -
        positions: [1.954989355354989, 0.24356691602066344, 1.218203873035374, -0.1274402173855089, 0.3404407093088011, 2.7335148417291024]
        velocities: [-0.47285358351270473, 0.05355781709191802, -0.21136168254752016, 0.042720188309108315, 0.19168443633832005, -0.3177103190318664]
        accelerations: [-1.0057383706771958, 0.11391507556927635, -0.449556821901979, 0.0908639250777112, 0.40770420148861797, -0.6757556033237845]
        effort: []
        time_from_start:
          secs: 0
          nsecs: 552823015
      -
        positions: [1.8021663394615532, 0.260876433000217, 1.1498932354564686, -0.113633347257902, 0.4023917966695852, 2.6308330585559165]
        velocities: [-0.765307758920758, 0.08668267387721376, -0.34208630585077826, 0.06914210384669317, 0.31023892280627474, -0.5142102771813788]
        accelerations: [-0.9467672086246705, 0.1072356999210169, -0.4231971950157713, 0.0855361465950437, 0.3837985902119062, -0.6361328800059017]
        effort: []
        time_from_start:
          secs: 0
          nsecs: 781167210
```

Thank you!