



# 机械臂规划Moveit

朱笑笑

2017年12月



上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

# 机械臂规划Moveit



- 机器人的运动规划的定义如下：在环境中找到一条符合约束条件的路径。约束条件可能包括障碍物、机器人的几何形状、速度限制等。


- 一般认为，运动规划是NP-hard问题，也即问题复杂性随着问题维度（机器人自由度）的增加而快速增加。

# 机械臂规划Moveit



- Moveit！是目前针对移动操作最先进的软件。结合了运动规划，操纵，三维感知，运动学，控制和导航的最新进展。它提供了一个易于使用的平台，开发先进的机器人应用程序，评估新的机器人设计和建筑集成的机器人产品它广泛应用于工业，商业，研发和其他领域。
- Moveit！是最广泛使用的开源软件的操作，并已被用于超过65个机器人



# 机械臂规划Moveit





1

Moveit系统结构

2

Moveit内部构成

3

Moveit规划测试



1

Moveit系统结构

2

Moveit内部构成

3

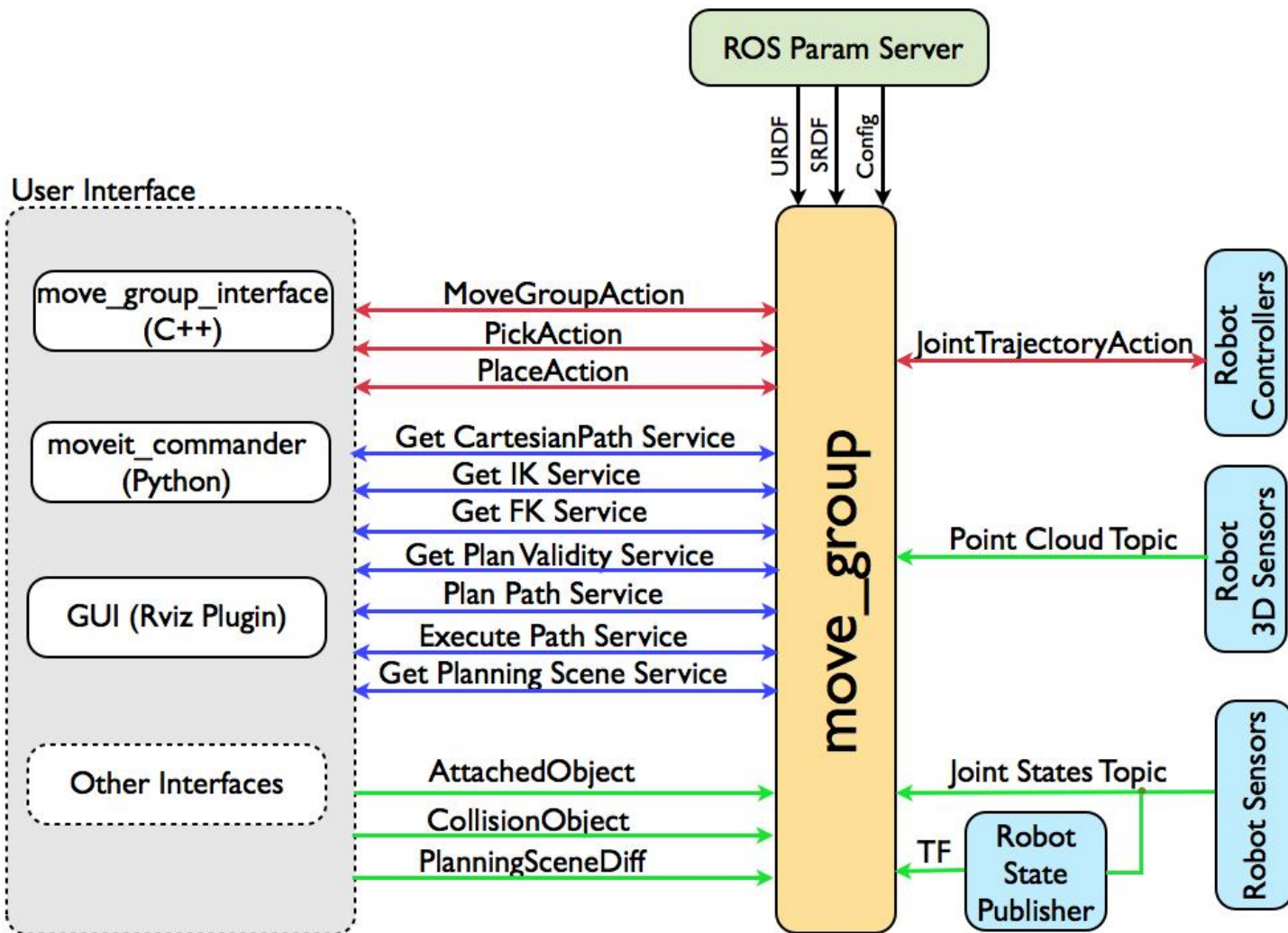
Moveit规划测试



上海交通大学

SHANGHAI JIAO TONG UNIVERSITY

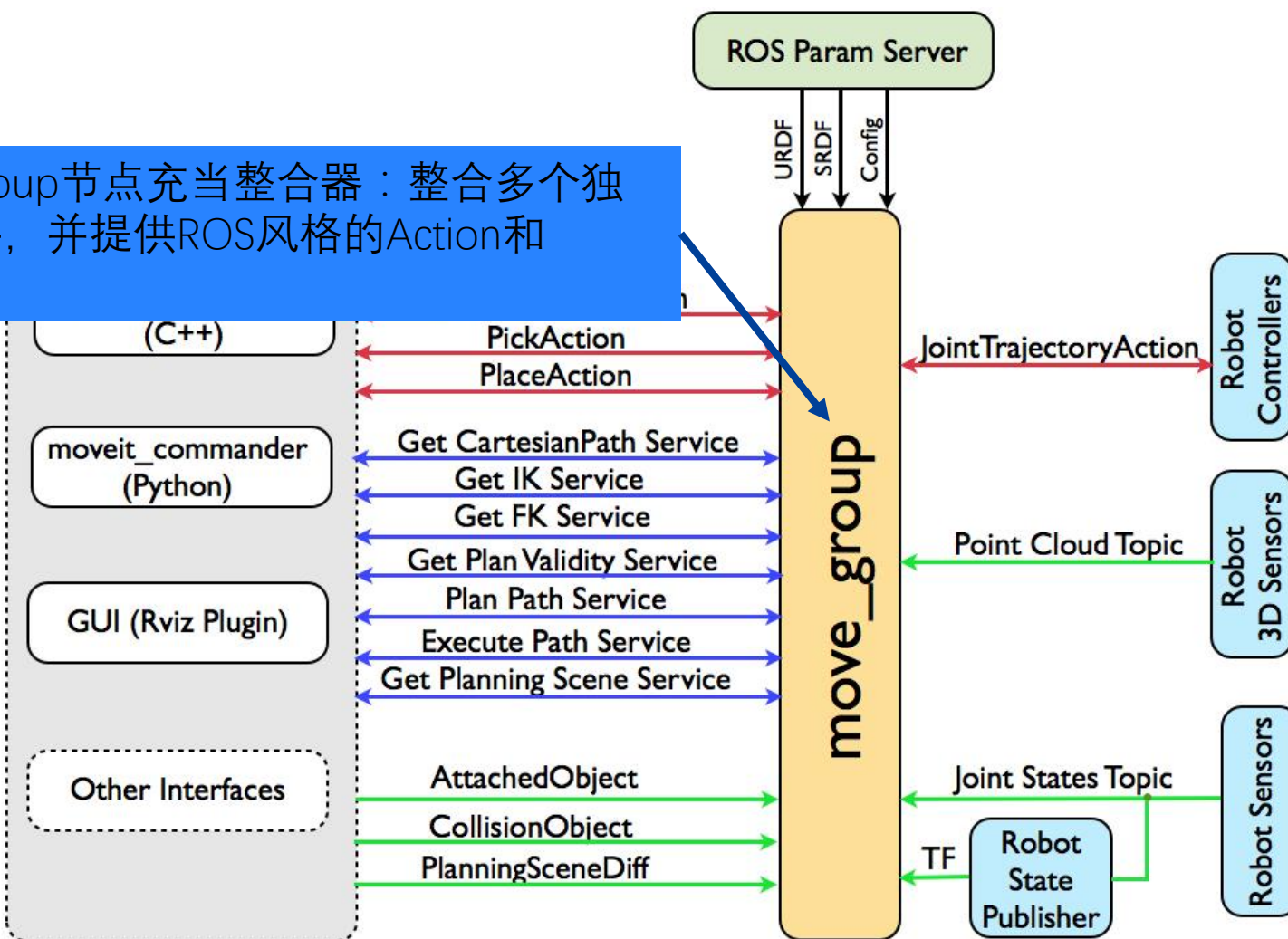
# Moveit系统结构



# Moveit系统结构

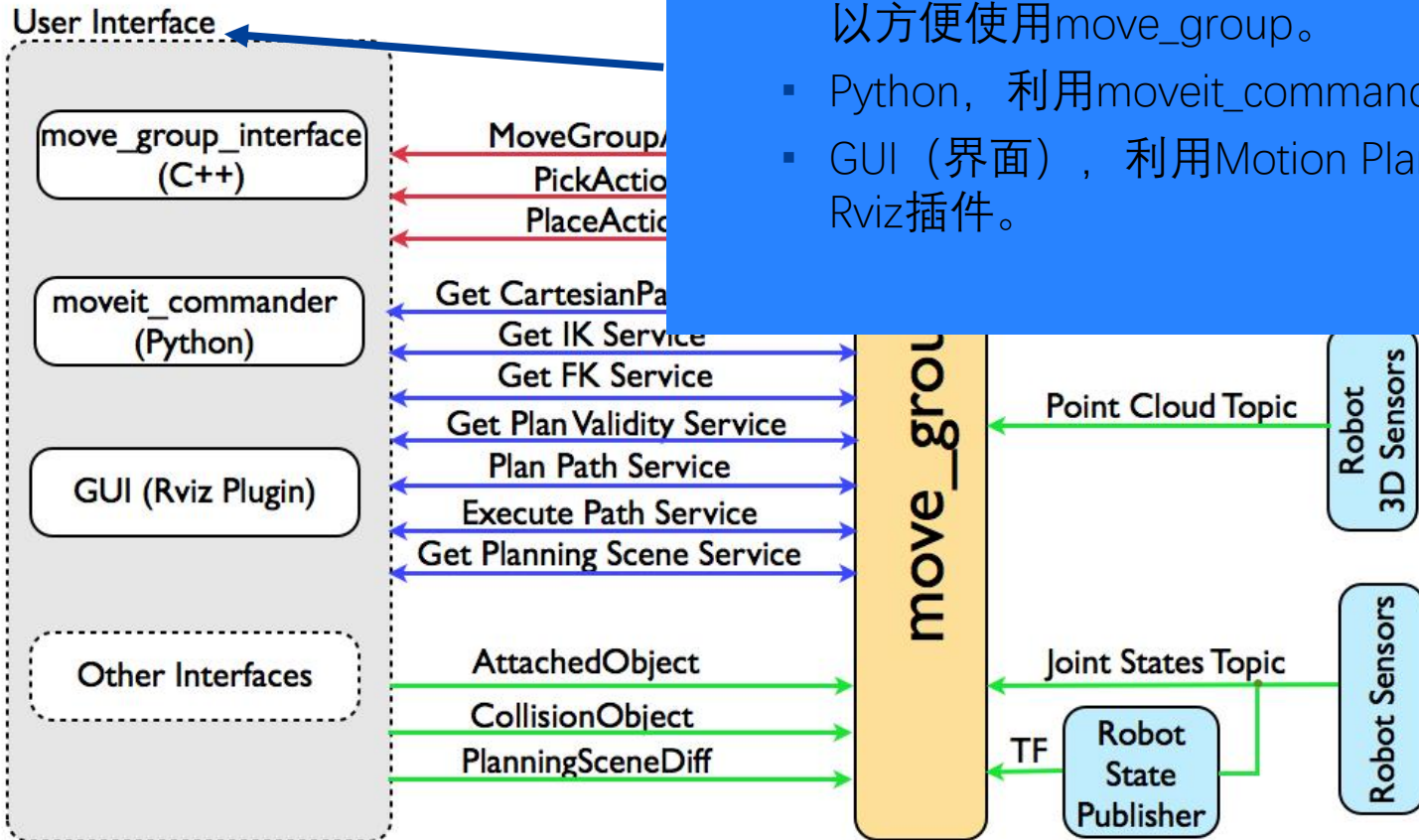


move\_group节点充当整合器：整合多个独立的组件，并提供ROS风格的Action和service



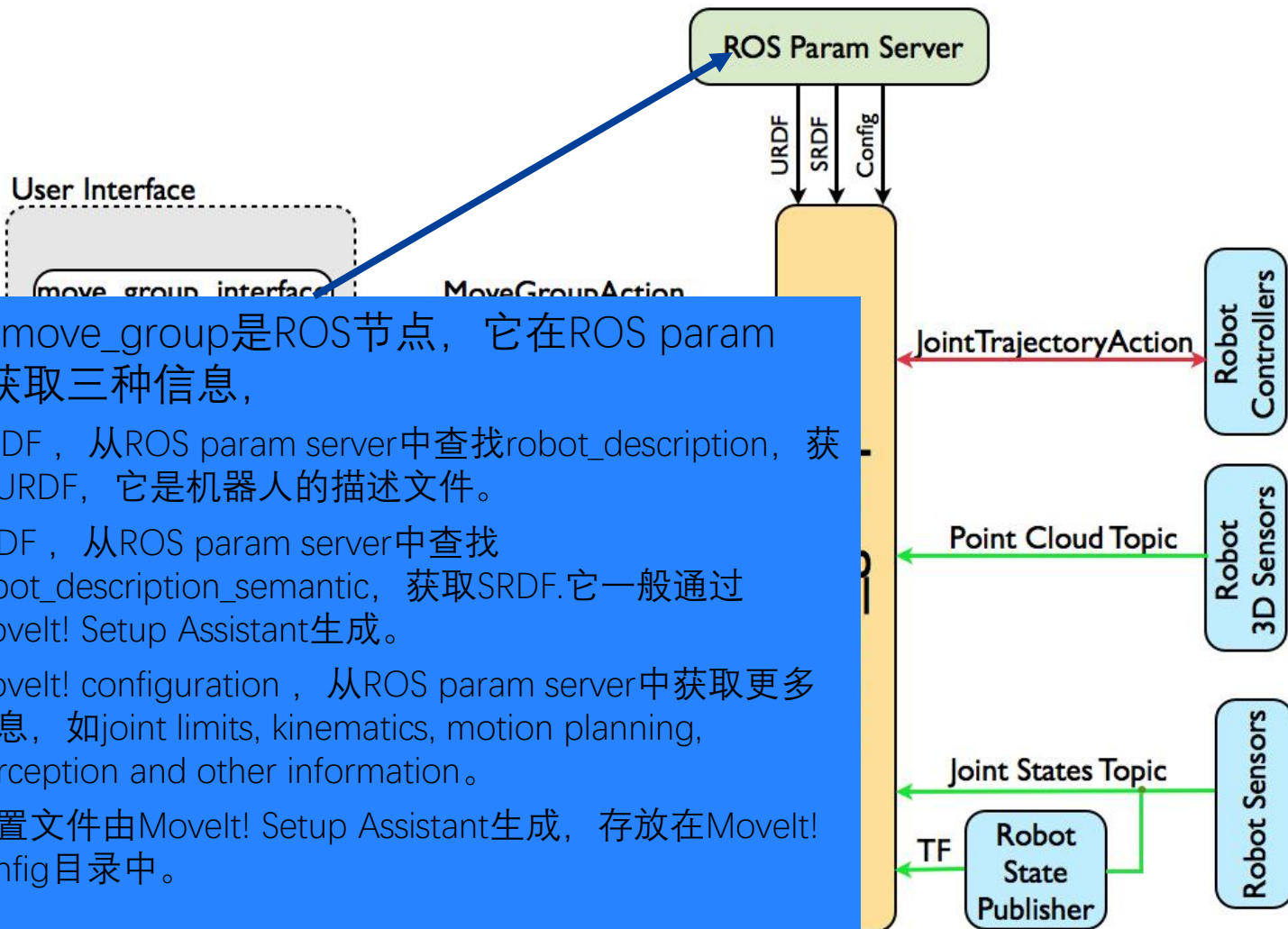


# Moveit系统结构



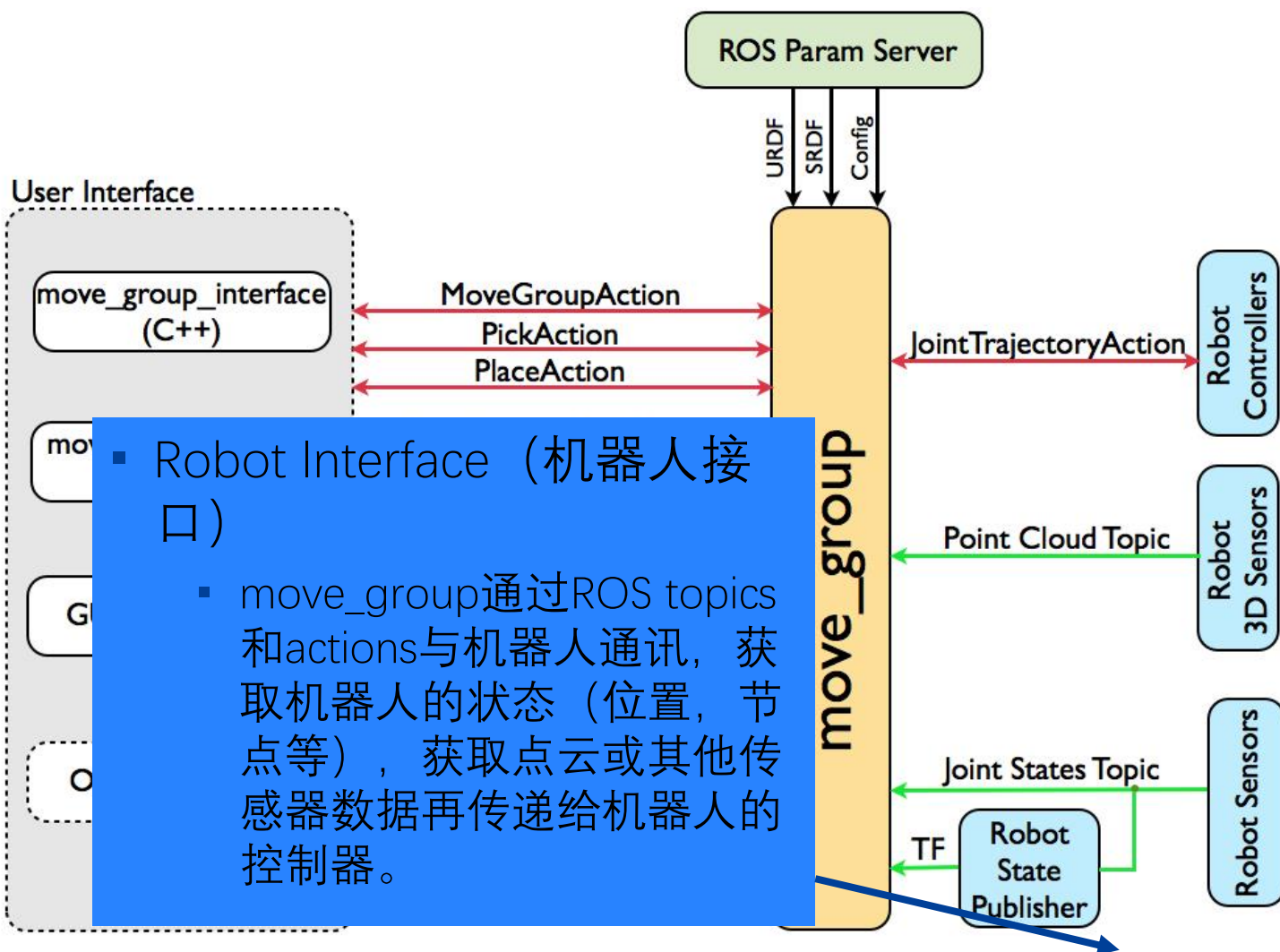
- User Interface（用户接口，三种接口可供调用）
  - C++，利用move\_group\_interface包可以方便使用move\_group。
  - Python，利用moveit\_commander包
  - GUI（界面），利用Motion Planning 的 Rviz插件。

# Moveit系统结构



- 配置，move\_group是ROS节点，它在ROS param server获取三种信息，
  - URDF，从ROS param server中查找robot\_description，获取URDF，它是机器人的描述文件。
  - SRDF，从ROS param server中查找robot\_description\_semantic，获取SRDF.它一般通过MoveIt! Setup Assistant生成。
  - MoveIt! configuration，从ROS param server中获取更多信息，如joint limits, kinematics, motion planning, perception and other information。
  - 配置文件由MoveIt! Setup Assistant生成，存放在MoveIt! config目录中。

# Moveit系统结构



# Moveit系统结构——RobotInterface



- Joint State Information （节点状态信息）
  - move\_group监听 /joint\_states 主题确定状态信息。例如：确定每个节点的位置。
  - Move\_group能够监听在这主题的多个发布者信息，即使是发布部分的信息（例如：独立的发布者可能是用于机械臂或移动机器人）。
  - move\_group不会建立自己的节点状态发布者。这就需要在每个机器人单独来建立。
- Transform Information （变换信息）
  - Move\_group通过ROS TF库来监视变换信息。这允许节点获取全局的姿态信息。
  - 例如：navigation包发布机器人的map frame和base frame到TF， move\_group可以使用TF找出这个变换信息，在内部使用。
  - 注意：Move\_group只是监听TF，你需要启动robot\_state\_publisher才能发布TF。
- Controller Interface （控制器接口）
  - 通过ROS的action接口， FollowJointTrajectoryAction接口来使用控制器。
  - 一个机器人的服务器服务于这个action-这个服务器不是有move\_group提供。
  - move\_group只会实例化客户端与机器人的控制器action服务器通讯。



# Moveit系统结构



- Planning Scene（规划场景）
  - move\_group使用规划场景监视器来维护规划场景。
  - 场景是世界的和机器人的状态的表现。
  - 机器人状态包含机器人刚性连接到机器人的所有物体。
  - 关于维护和更新规划场景的体系结构的详细信息在下面的规划场景部分中描述。
- Extensible Capabilities（可扩展能力）
  - move\_group的结构被设计成容易扩展，独立的能力如抓放，运动学，运动规划。
  - 扩展自公共类，但实际作为独立的插件运行。
  - 插件可经由一系列的ROS yaml parameters 和ROS pluginlib库配置。

1

Moveit系统结构

2

Moveit内部构成

3

Moveit规划测试



上海交通大学

SHANGHAI JIAO TONG UNIVERSITY



# Moveit内部构成



move\_group

motion\_planner

Kinematics

Planning Scene Monitor

world geometry  
monitor

scene monitor

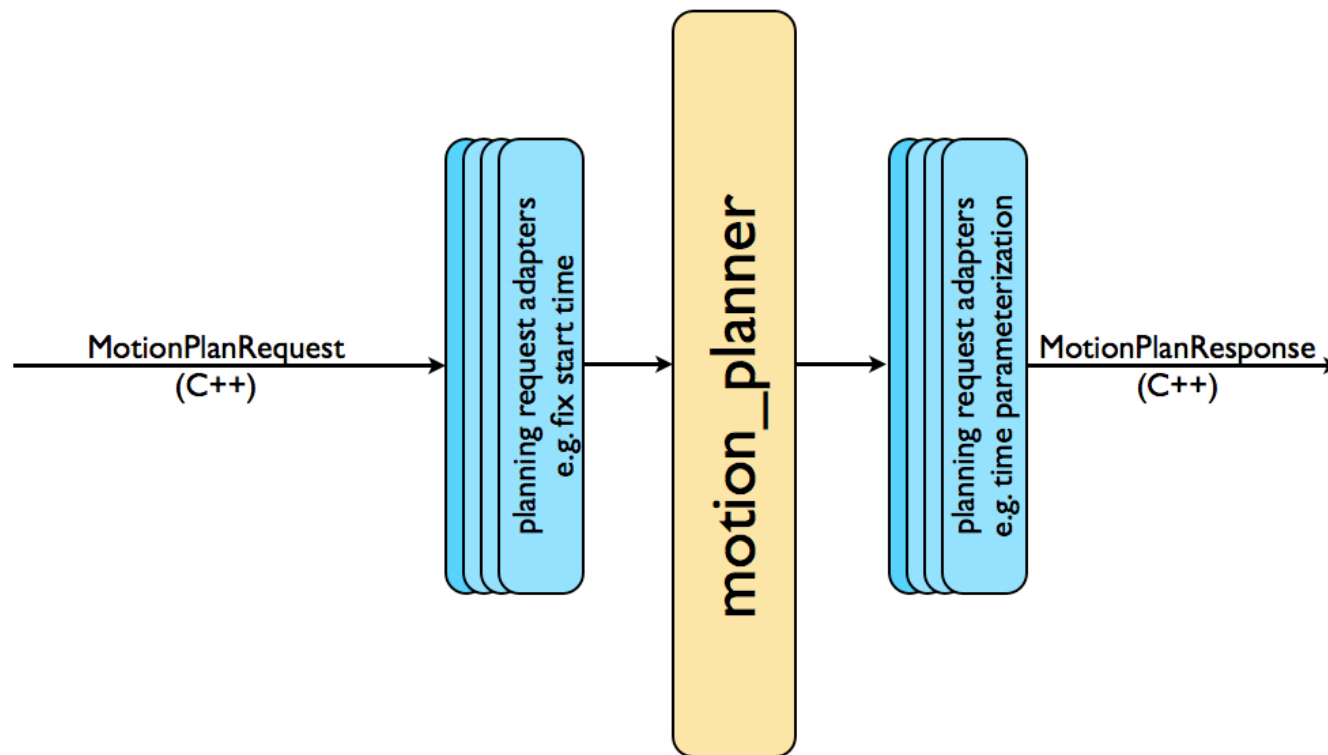
state monitor

Collision Checking

Trajectory Processing

# 1、运动规划插件

- 运动规划管道





# 1、运动规划插件



## ■ 运动规划管道

- 完整的运动规划管道链整合规划规划器与多个叫规划请求适配器的组件。
- 规划请求适配器允许规划请求预处理和规划反馈后处理。
- 预处理在某些情况有用，例如：机器人的起始状态稍微超出关节限制之外的情况。
- 后处理需要处理几个操作，例如：转换生成的路径为带时间参数的轨迹。
- MoveIt提供一系列默认的运动规划器适配器，每个都执行一个指定的功能。

# 1、运动规划插件



## ■ 运动规划管道

- 常用运动规划适配器：FixStartStateBounds, 这个适配器修复在URDF文件里面描述的关节限制的开始状态。用于仿真器中，当机器人配置不正确的时候。当有一个或多个关节稍微超出限制，机器人可能会结束。在这种情况下，运动规划就不能正常执行，因为它认为已经超出了关节限制。

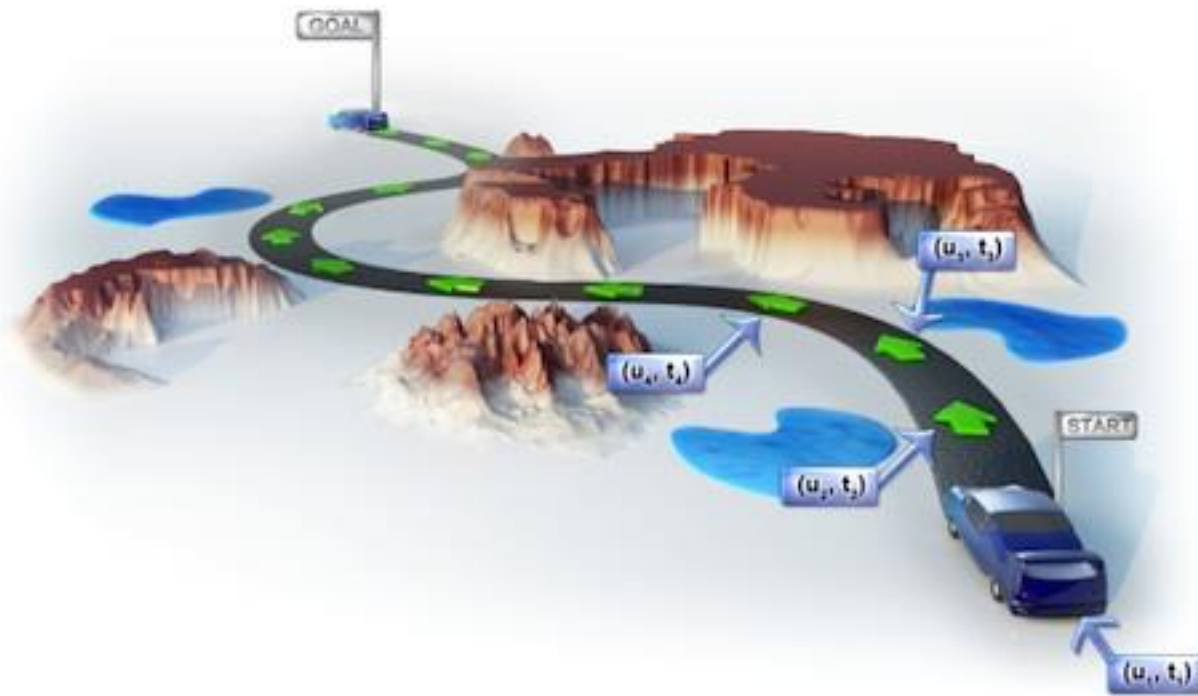
FixStartStateBounds请求适配器会修复开始状态到设置的关节限制内。但这不是每一次都是合适的解决方案。例如有多少关节超出限制。适配器参数则指定有多少关节超出限制才启用修复。

- FixWorkspaceBounds, 这个适配器会为规划指定一个默认的工作空间，一个立方体的大小为10米×10米×10米。如果规划请求的规划器没有填充这些区域，将会指定工作空间。

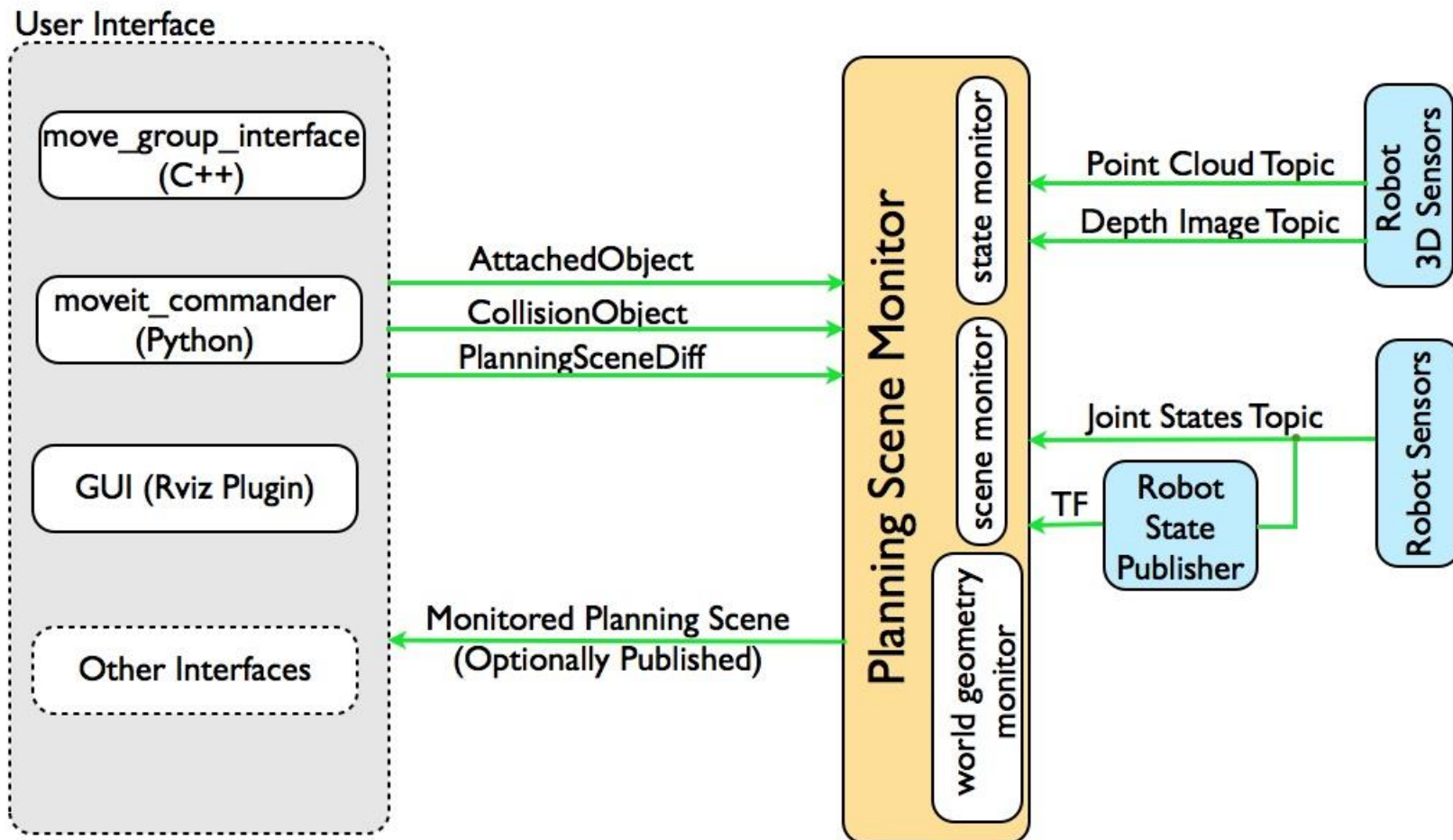
# 1、运动规划插件



- OMPL (Open Motion Planning Library) 是一个开源的运动规划库，主要是执行随机规划器。MoveIt直接整合OMPL，使用其库里的运动规划器作为主要/默认的一套规划器。在OMPL规划器是抽象的，例如：OMPL没有机器人的概念。MoveIt!配置OMPL，提供一个后端处理，用于解决机器人的问题。



## 2、规划场景





## 2、规划场景



- 规划场景，用于显示机器人的世界，同时保存机器人自己的状态。它由Move\_group节点内的规划场景监视器来维护。  
规划场景监视器监听：
  - State Information（状态的信息）：joint\_states 主题
  - Sensor Information（传感器的信息）：using the world geometry monitor described below
  - World geometry information（世界的几何图形信息）：from user input on the planning\_scene topic (as a planning scene diff).
- World Geometry Monitor（**世界几何图形监视器**），它通过来自机器人的传感器信息和来自用户的输入建立世界几何图形。它使用occupancy map monitor（occupancy地图监视器）建立围绕机器人的3D感知环境和通过planning\_scene主题中附带的参数来增加对象的信息。

## 2、规划场景

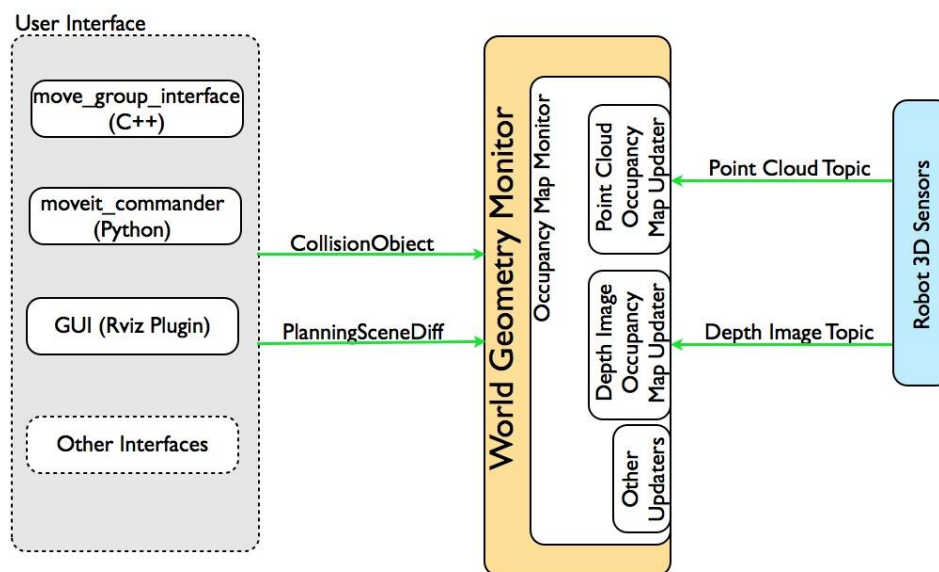


### ■ 3D Perception (3D感知)

在MoveIt, 3D感知是由occupancy map monitor处理, 它使用插件结构处理不同的传感器输入。MoveIt有两个内置支持可以处理两种输入:

- Point clouds: handled by the point cloud occupancy map updater plugin
- Depth images: handled by the depth image occupancy map updater plugin

- 你可为occupancy map monitor编写自己的插件。



### 3、运动学



- The Kinematics Plugin（运动学插件）， MoveIt!使用插件结构，尤其是允许用户编写自己的逆运动学算法。 Forward kinematics（正向运动学） and finding jacobians（查找雅可比矩阵） 被整合到自己的 RobotState类。默认逆运动学插件配置使用KDL numerical jacobian-based solver.由MoveIt! Setup Assistant自动配置。
- IKFast Plugin（IKFast插件）， 通常，用户可以选择执行自己的运动学求解器，例如PR2的有自己的运动学求解器。要实现这样的求解的一种流行的方法是使用ikfast包产生的需要与您的特定工作的机器人的C++代码。

## 4、碰撞检测



- Collision Checking（冲突检测），在规划场景中，冲突检测通过 CollisionWorld 对象来配置，由 FCL 包（主要的 CC 库）来执行。
- Collision Objects（冲突对象），MoveIt 支持不同类型对象的冲突检测。
  - Meshes（网格）
  - Primitive Shapes（基本形状） - 例如：boxes（箱），cylinders（圆柱），cones（圆锥），spheres（球） and planes（平面）
  - Octomap - Octomap 对象能直接用于冲突检测
- Allowed Collision Matrix (ACM)（免检冲突矩阵），在运动规划里，冲突检测会耗费甚至达到90%的计算资源。ACM编码需要检测的对象间的对应关系（机器人的或世界的）。如果在ACM关联两对象的值为1，那就不需要检测，这情况就比如两个对象相隔很远，永远不会发生碰撞。



## 5、轨迹处理



- Trajectory Processing (轨迹处理)
- Time parameterization (时间参数化)，运动规划器一般只会生成路径，这个路径不带时间信息。MoveIt包含轨迹处理程序。它对结合路径和时间参数化的关节限制的速度和加速度来生成轨迹。这些限制是在joint\_limits.yaml中为每个机器人指定的。

1

Moveit系统结构

2

Moveit内部构成

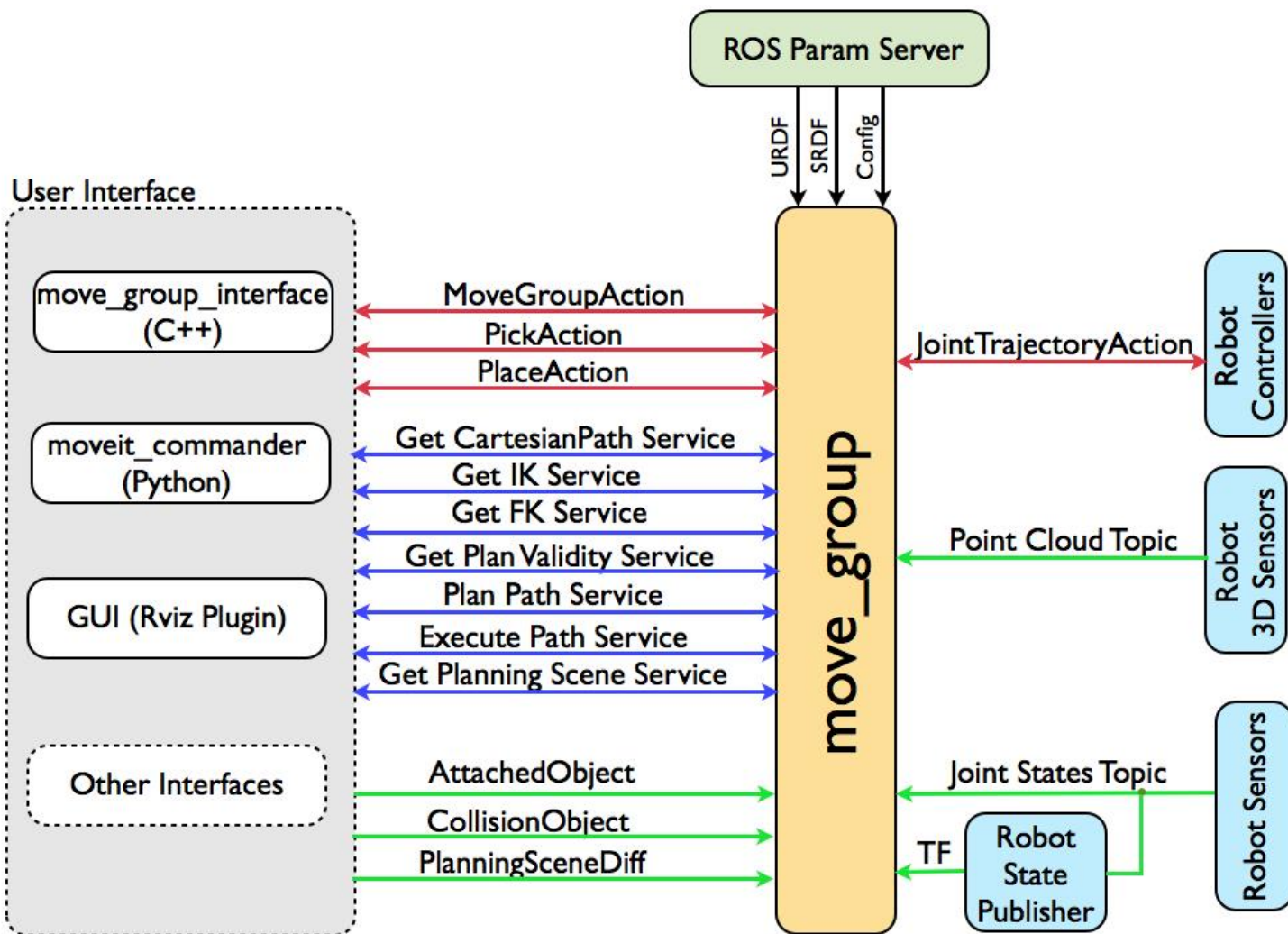
3

Moveit规划测试



# Moveit规划测试

编程  
图形化



# Moveit规划测试

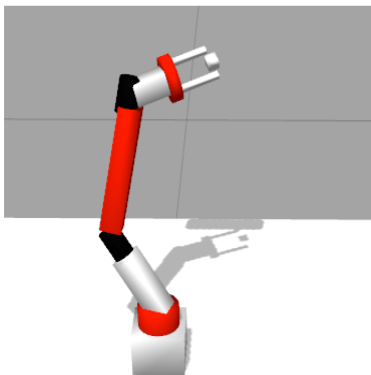


- 安装moveit相关包

`sudo apt-get install ros-kinetic-moveit*`

1、利用**moveit\_setup\_assistant** 包为机器人配置moveit的相关文件

2、利用RVIZ可视化界面进行轨迹规划

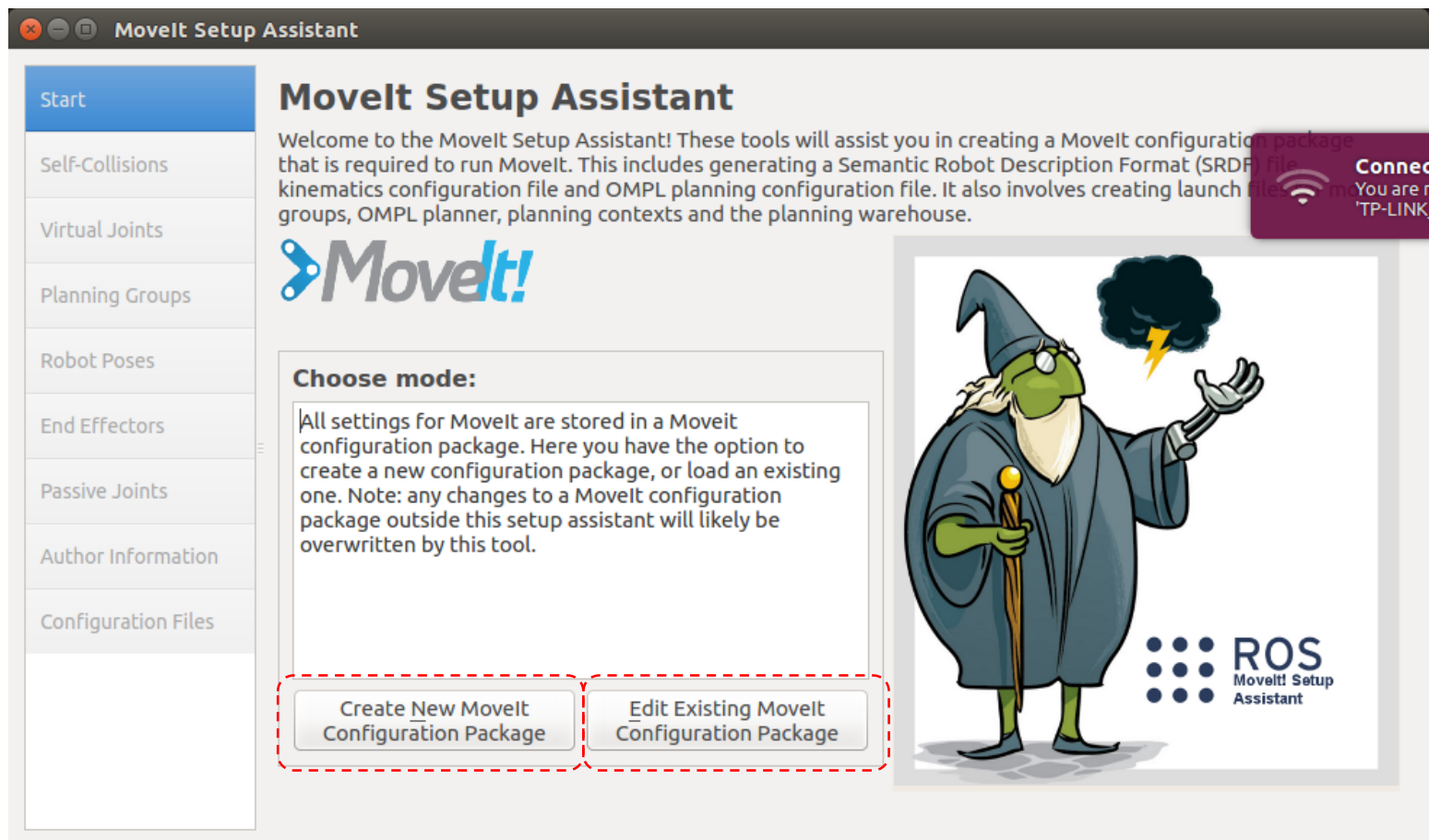


Chapter8/arm\_description

# moveit\_setup\_assistant配置



\$ roslaunch moveit\_setup\_assistant setup\_assistant.launch

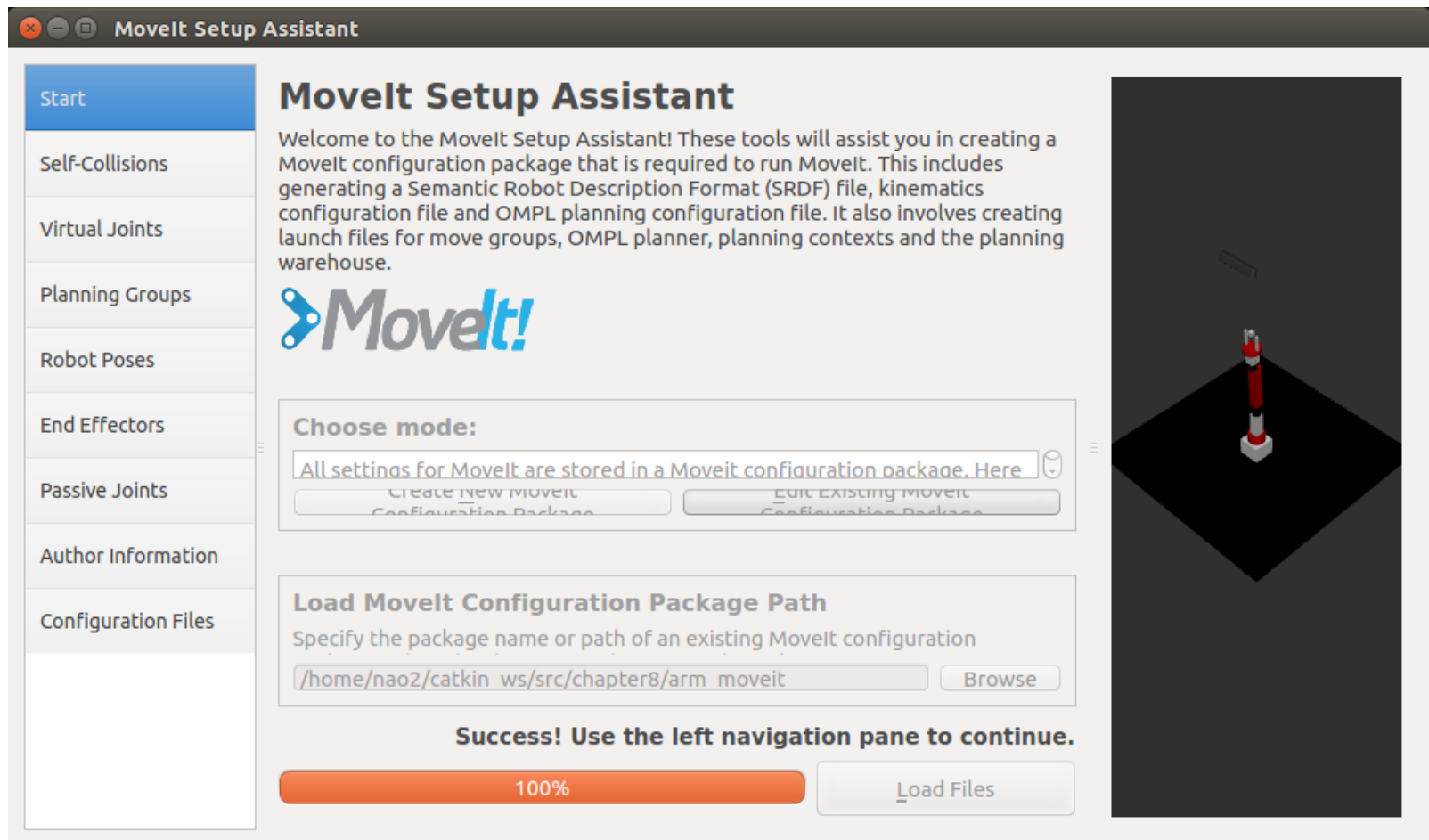




# moveit\_setup\_assistant配置

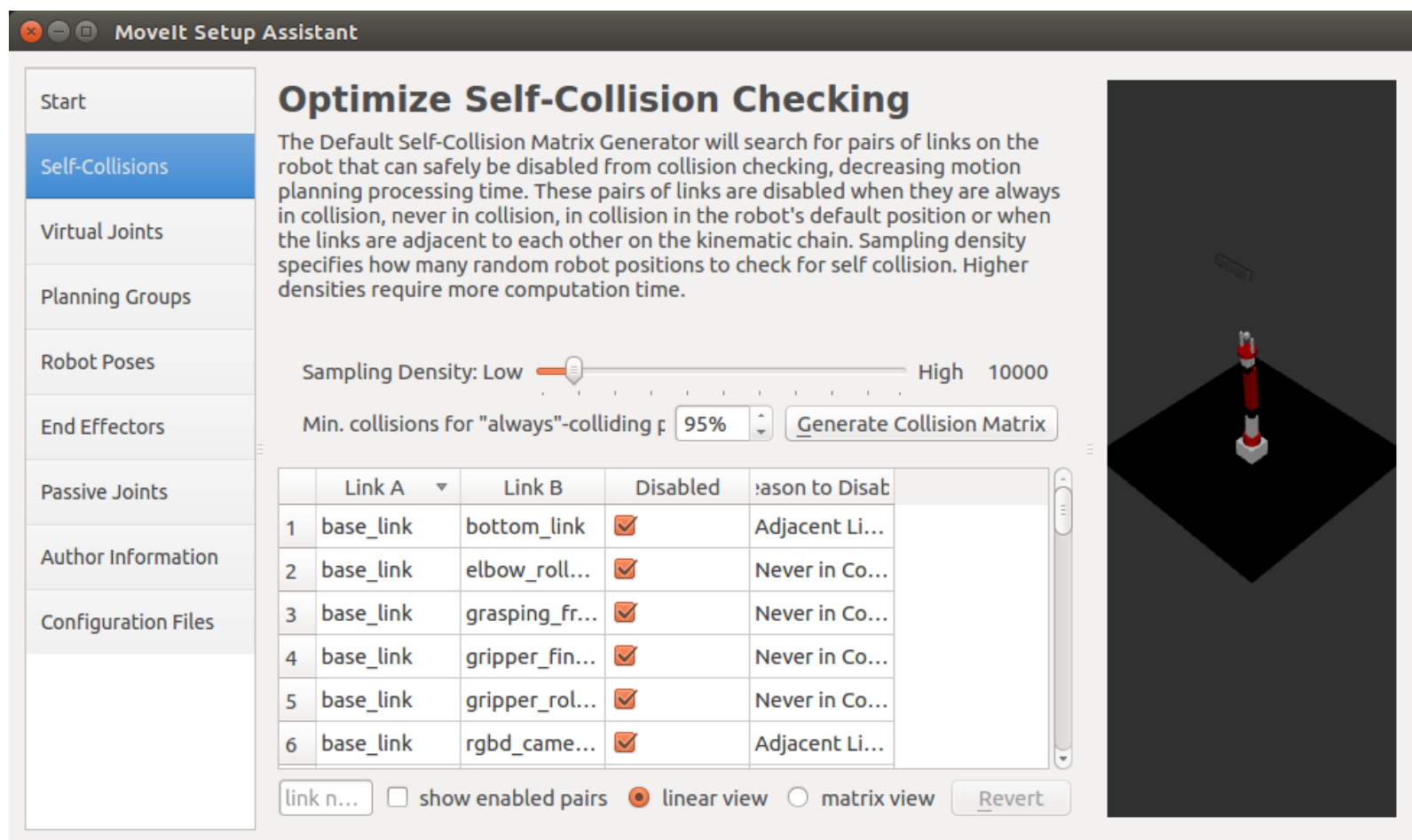


载入机器人的urdf模型



# moveit\_setup\_assistant配置

## 自身免碰撞设置



**MoveIt Setup Assistant**

**Optimize Self-Collision Checking**

The Default Self-Collision Matrix Generator will search for pairs of links on the robot that can safely be disabled from collision checking, decreasing motion planning processing time. These pairs of links 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 random robot positions to check for self collision. Higher densities require more computation time.

Sampling Density: Low  High 10000

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

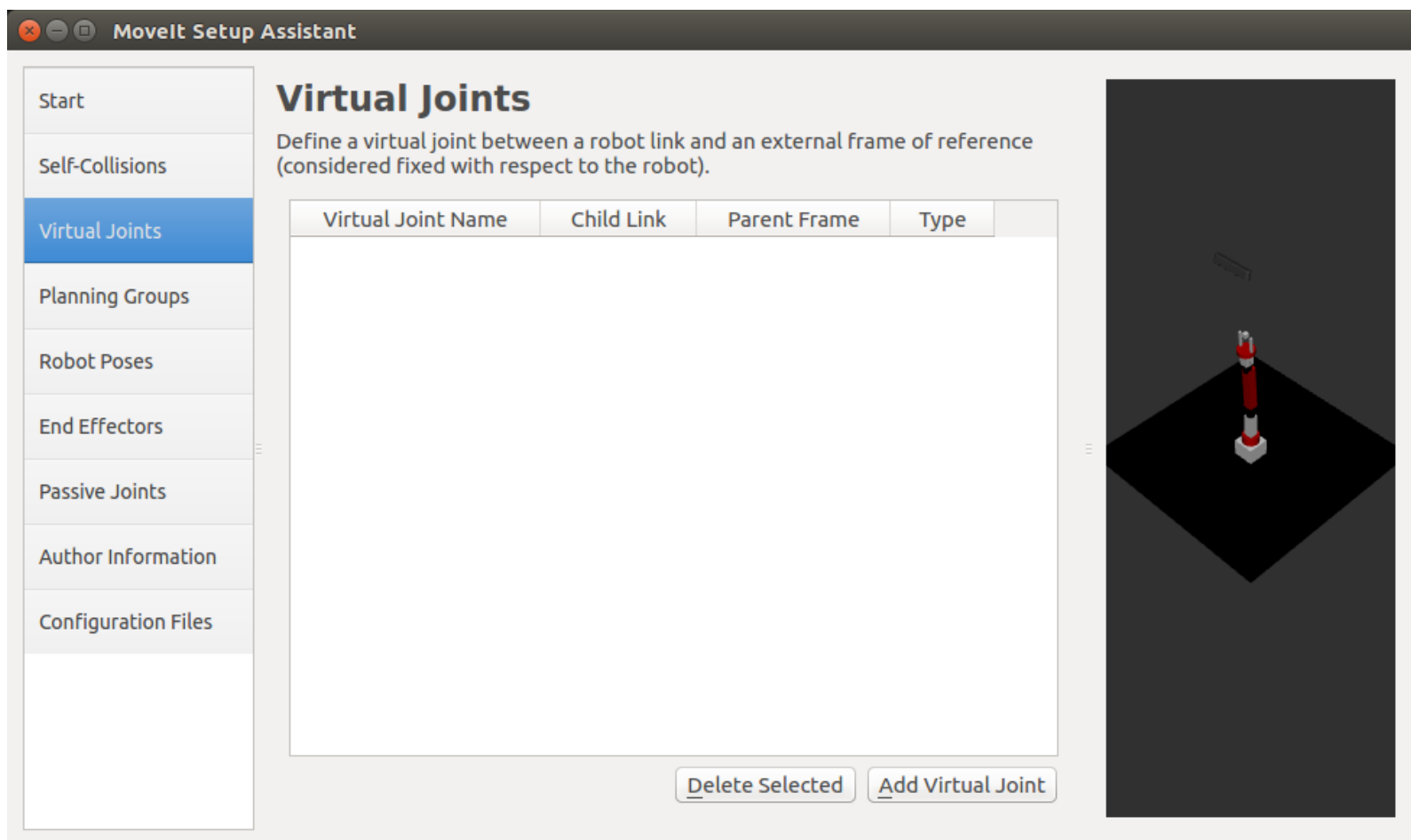
	Link A	Link B	Disabled	Reason to Disab
1	base_link	bottom_link	<input checked="" type="checkbox"/>	Adjacent Li...
2	base_link	elbow_roll...	<input checked="" type="checkbox"/>	Never in Co...
3	base_link	grasping_fr...	<input checked="" type="checkbox"/>	Never in Co...
4	base_link	gripper_fin...	<input checked="" type="checkbox"/>	Never in Co...
5	base_link	gripper_rol...	<input checked="" type="checkbox"/>	Never in Co...
6	base_link	rgb_d_came...	<input checked="" type="checkbox"/>	Adjacent Li...

link n... ☐ show enabled pairs ☒ linear view ☐ matrix view

# moveit\_setup\_assistant配置



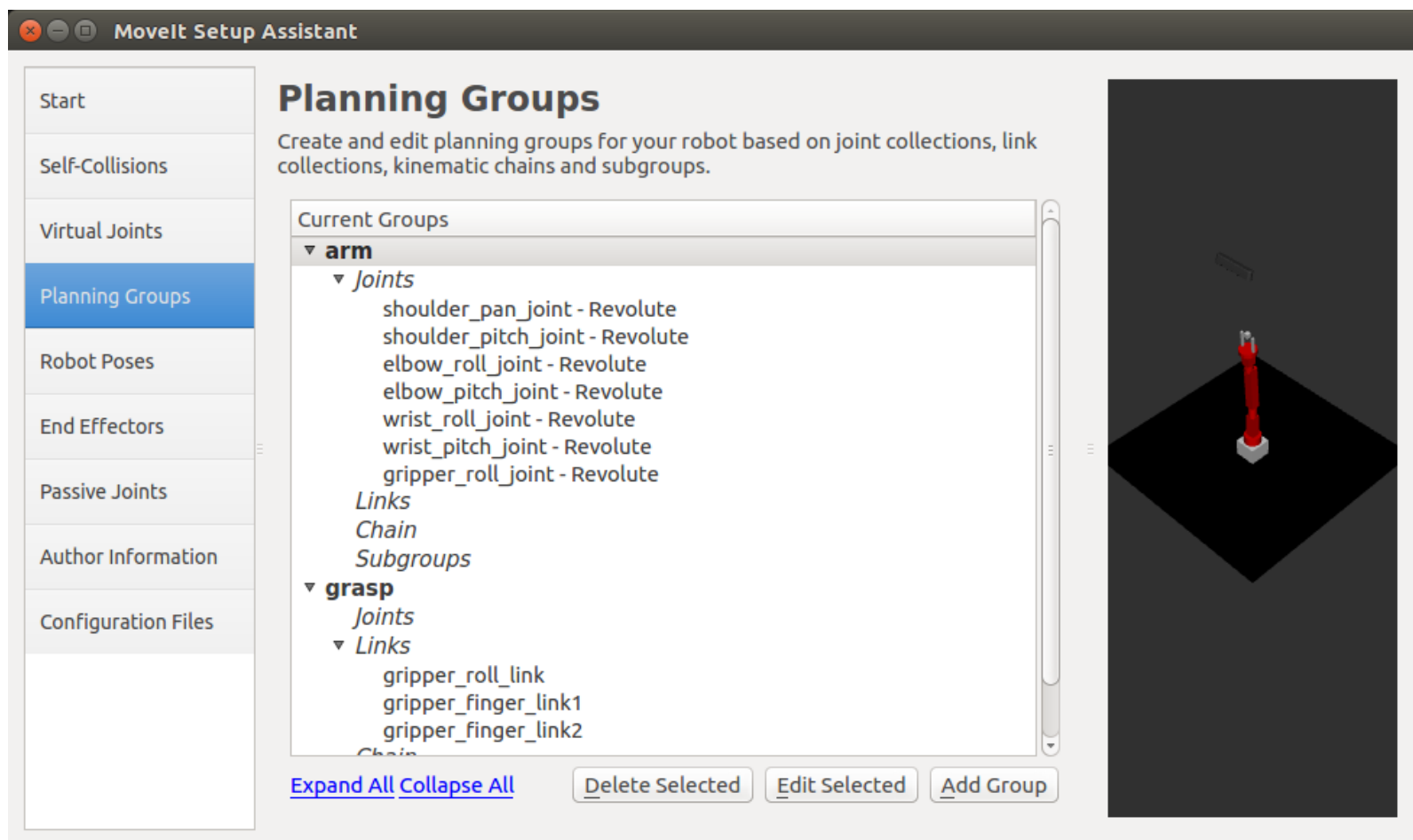
## 虚拟关节



# moveit\_setup\_assistant配置



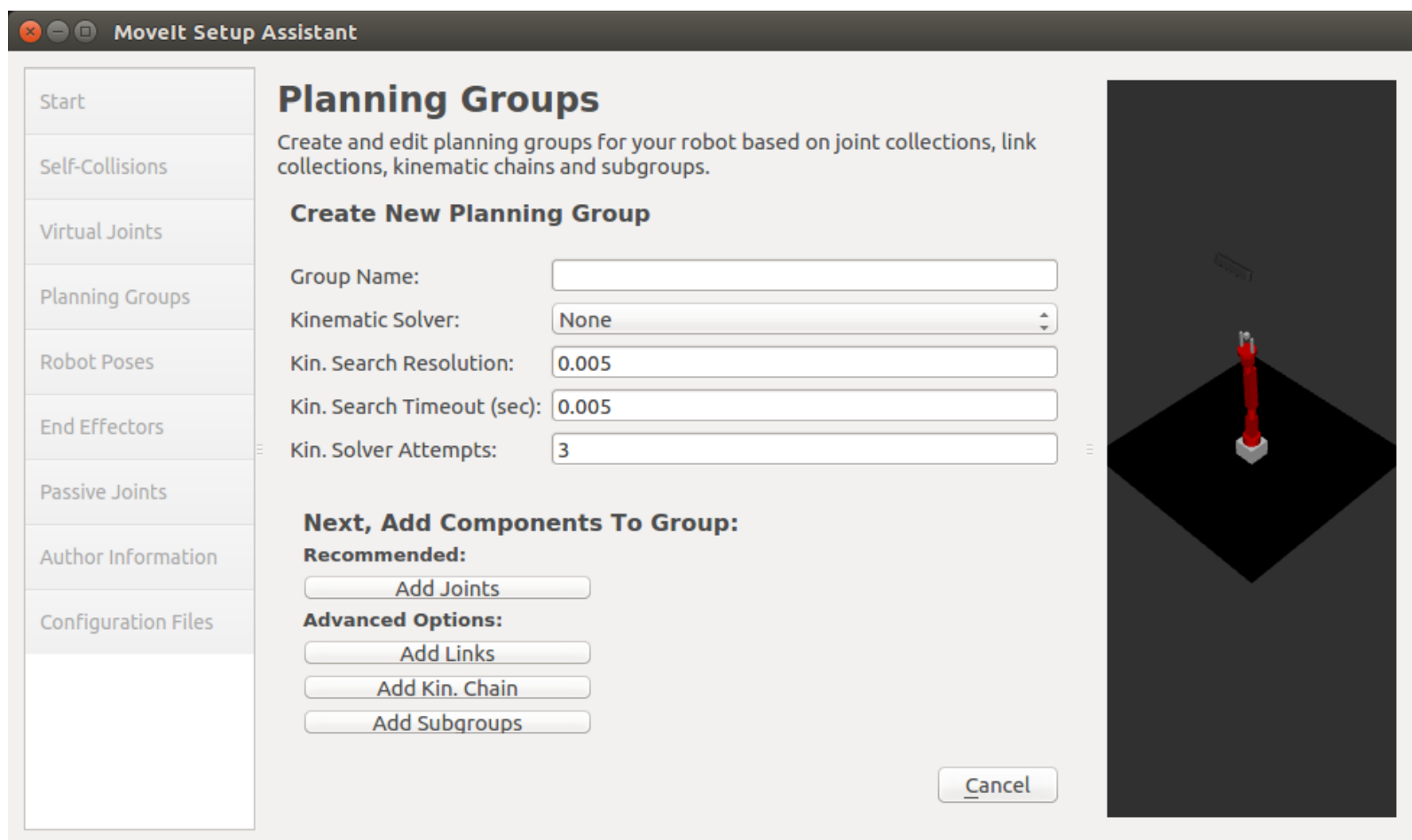
## 规划组



# moveit\_setup\_assistant配置



## 规划组

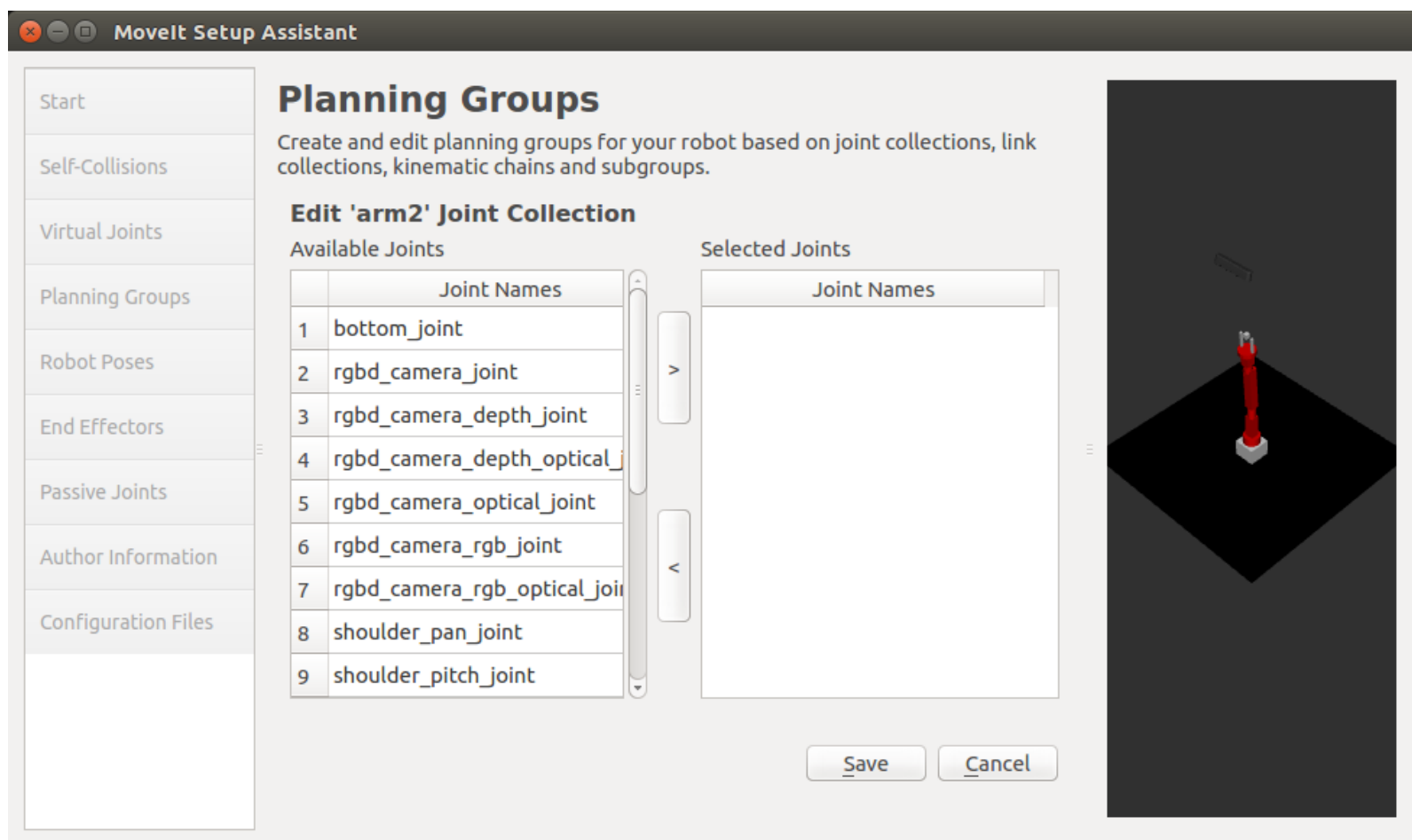




# moveit\_setup\_assistant配置



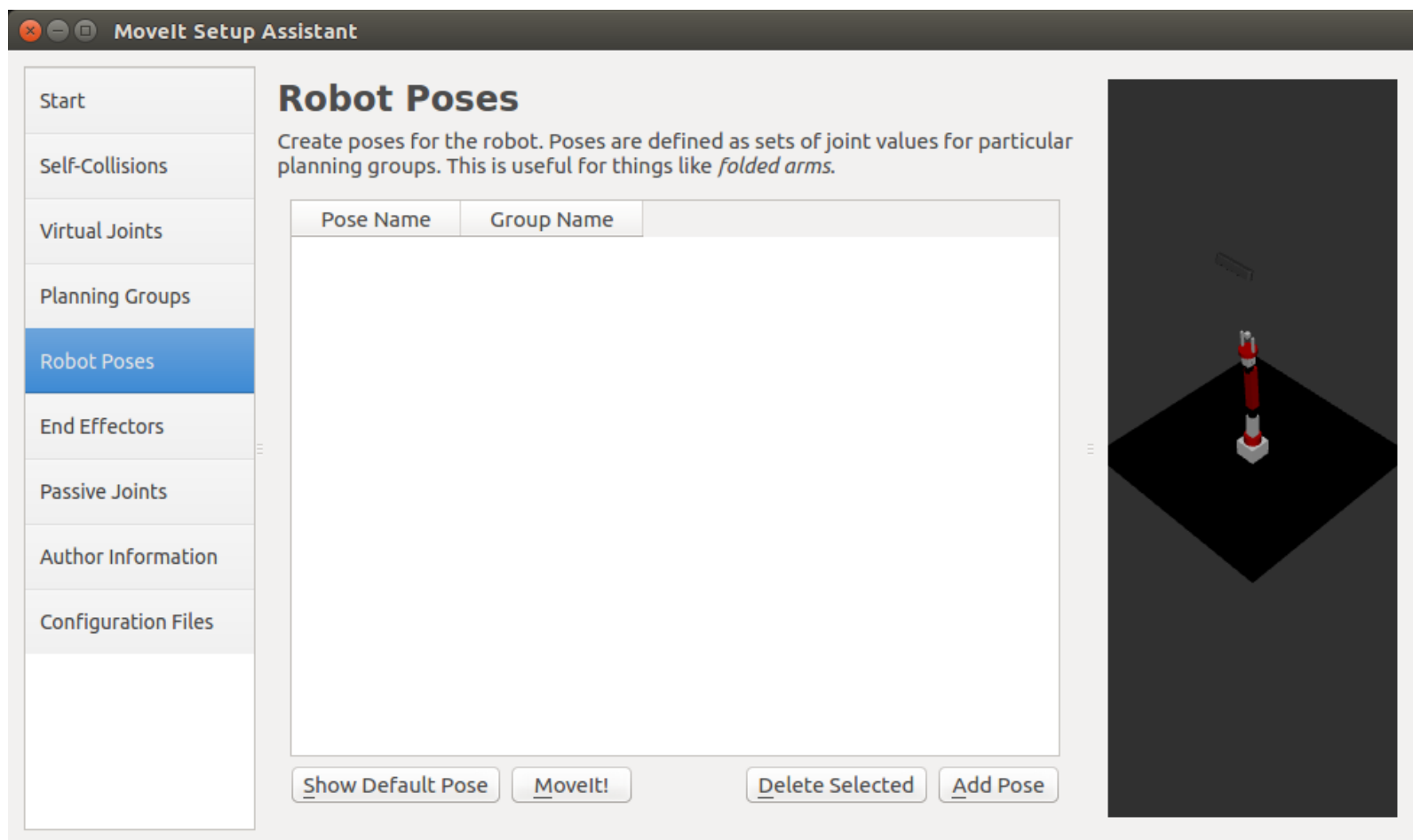
## 规划组



# moveit\_setup\_assistant配置



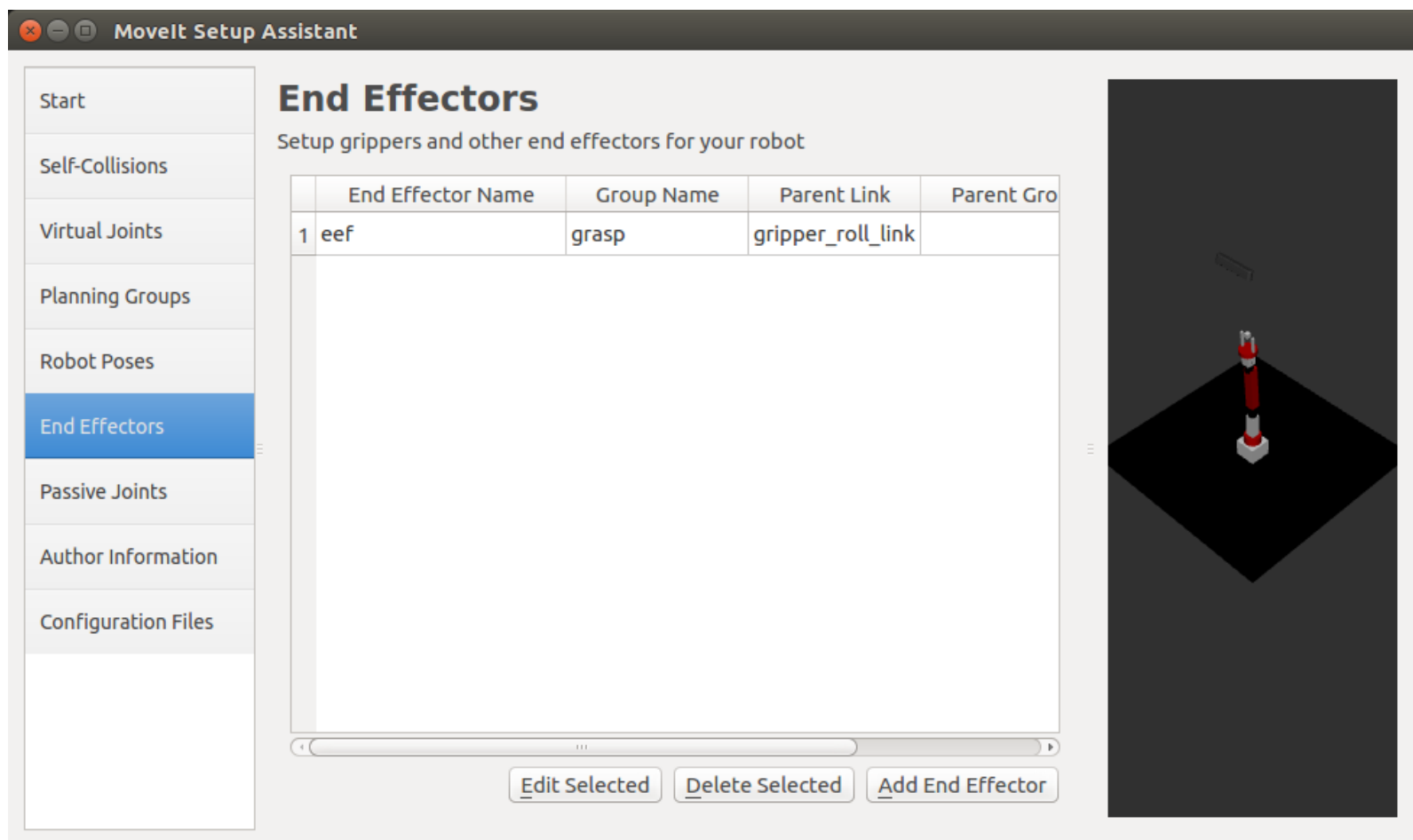
## 机器人预设姿态



# moveit\_setup\_assistant配置



## 末端工具坐标系



# moveit\_setup\_assistant配置



## 末端工具坐标系

The screenshot shows the 'MoveIt Setup Assistant' window with the 'End Effectors' tab selected. The left sidebar contains a list of configuration steps: Start, Self-Collisions, Virtual Joints, Planning Groups, Robot Poses, End Effectors (selected), Passive Joints, Author Information, and Configuration Files. The main area is titled 'End Effectors' with the subtitle 'Setup grippers and other end effectors for your robot'. It contains four input fields: 'End Effector Name' (empty), 'End Effector Group' (set to 'arm'), 'Parent Link (usually part of the arm):' (set to 'base\_link'), and 'Parent Group (optional):' (empty). At the bottom right are 'Save' and 'Cancel' buttons. On the far right, a 3D visualization shows a red and white robotic arm model on a black platform.

MoveIt Setup Assistant

**End Effectors**  
Setup grippers and other end effectors for your robot

End Effector Name:

End Effector Group:

Parent Link (usually part of the arm):

Parent Group (optional):

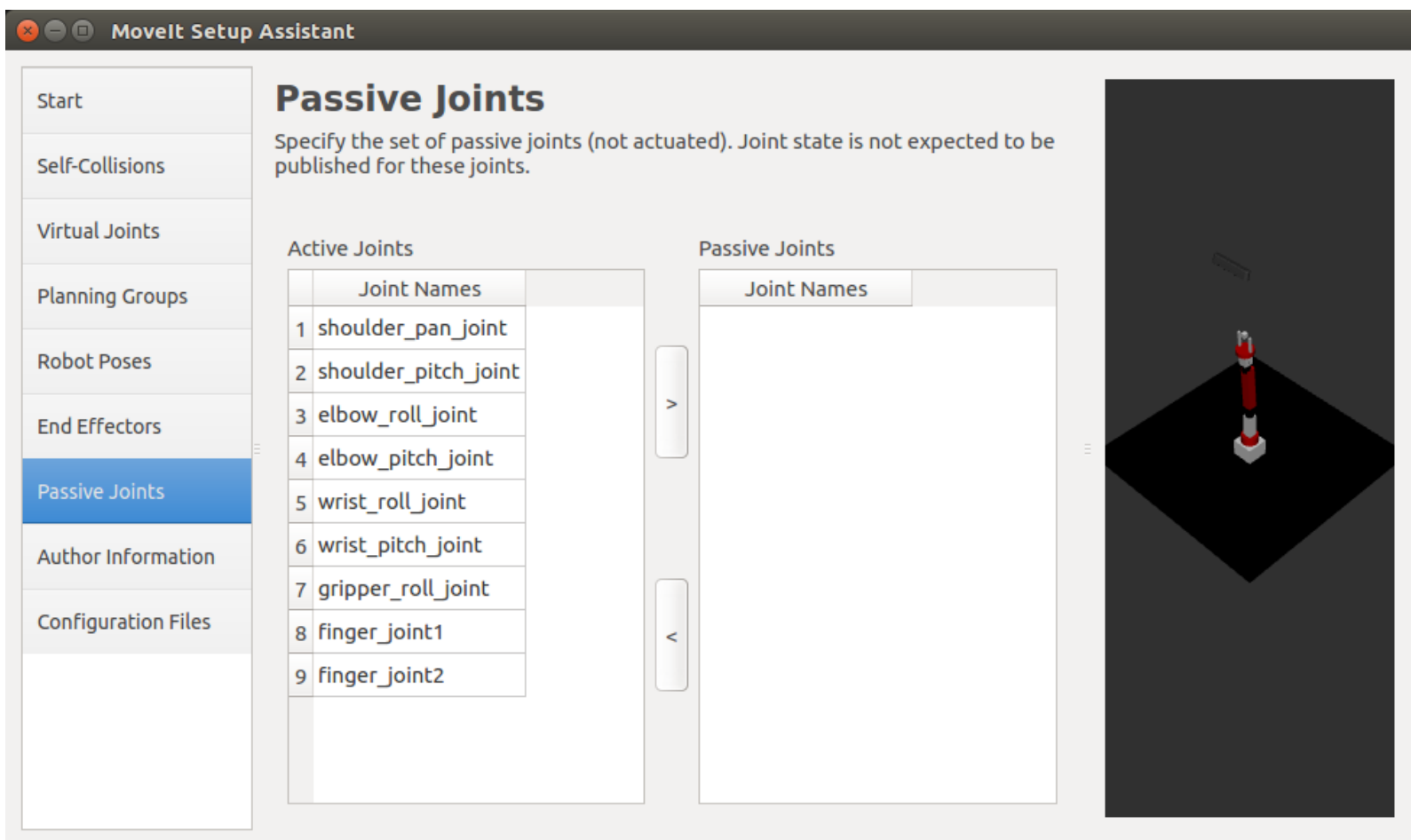
Start  
Self-Collisions  
Virtual Joints  
Planning Groups  
Robot Poses  
End Effectors  
Passive Joints  
Author Information  
Configuration Files

Save Cancel

# moveit\_setup\_assistant配置



## 被动关节



**MoveIt Setup Assistant**

**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	shoulder_pan_joint
2	shoulder_pitch_joint
3	elbow_roll_joint
4	elbow_pitch_joint
5	wrist_roll_joint
6	wrist_pitch_joint
7	gripper_roll_joint
8	finger_joint1
9	finger_joint2

**Passive Joints**

Joint Names
-------------

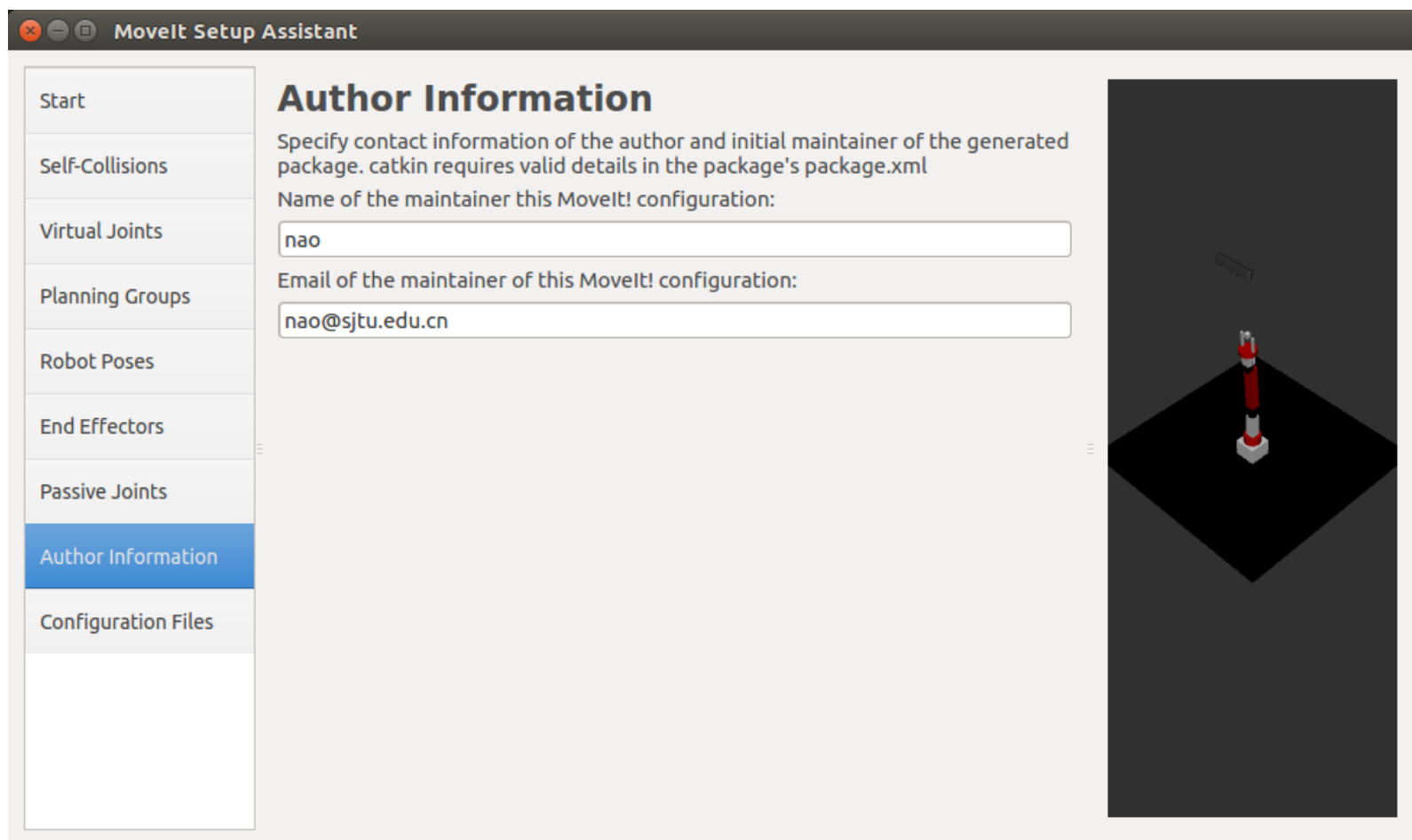
3D Visualization of a robot arm.



# moveit\_setup\_assistant配置



## 作者信息



**MoveIt Setup Assistant**

Start

Self-Collisions

Virtual Joints

Planning Groups

Robot Poses

End Effectors

Passive Joints

**Author Information**

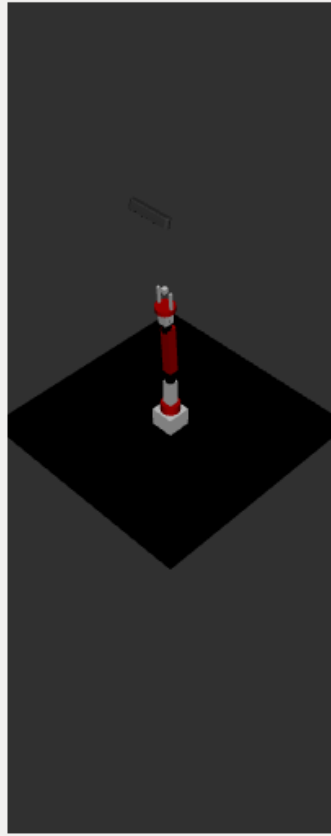
Configuration Files

**Author Information**

Specify 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 this MoveIt! configuration:

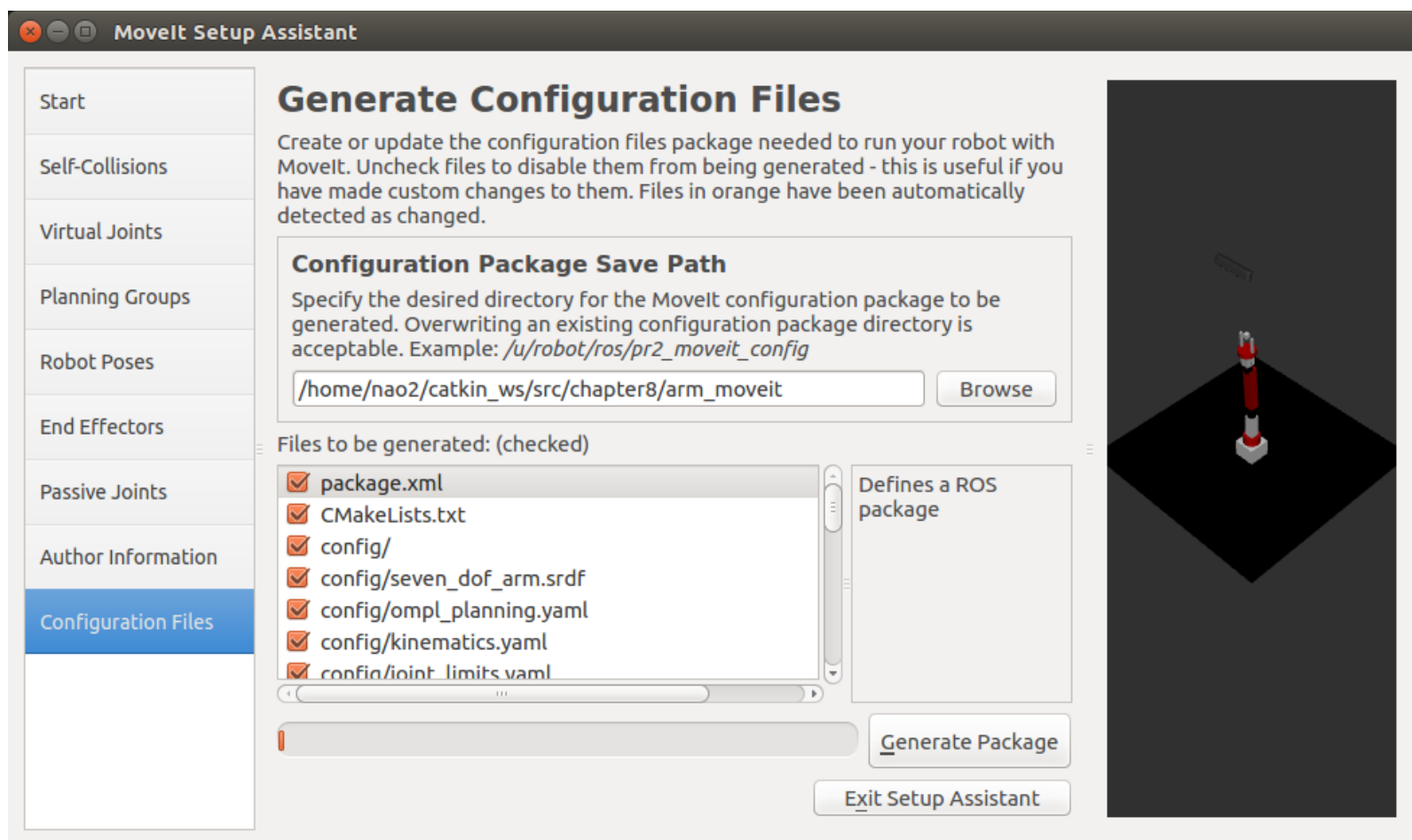
Email of the maintainer of this MoveIt! configuration:



# moveit\_setup\_assistant配置



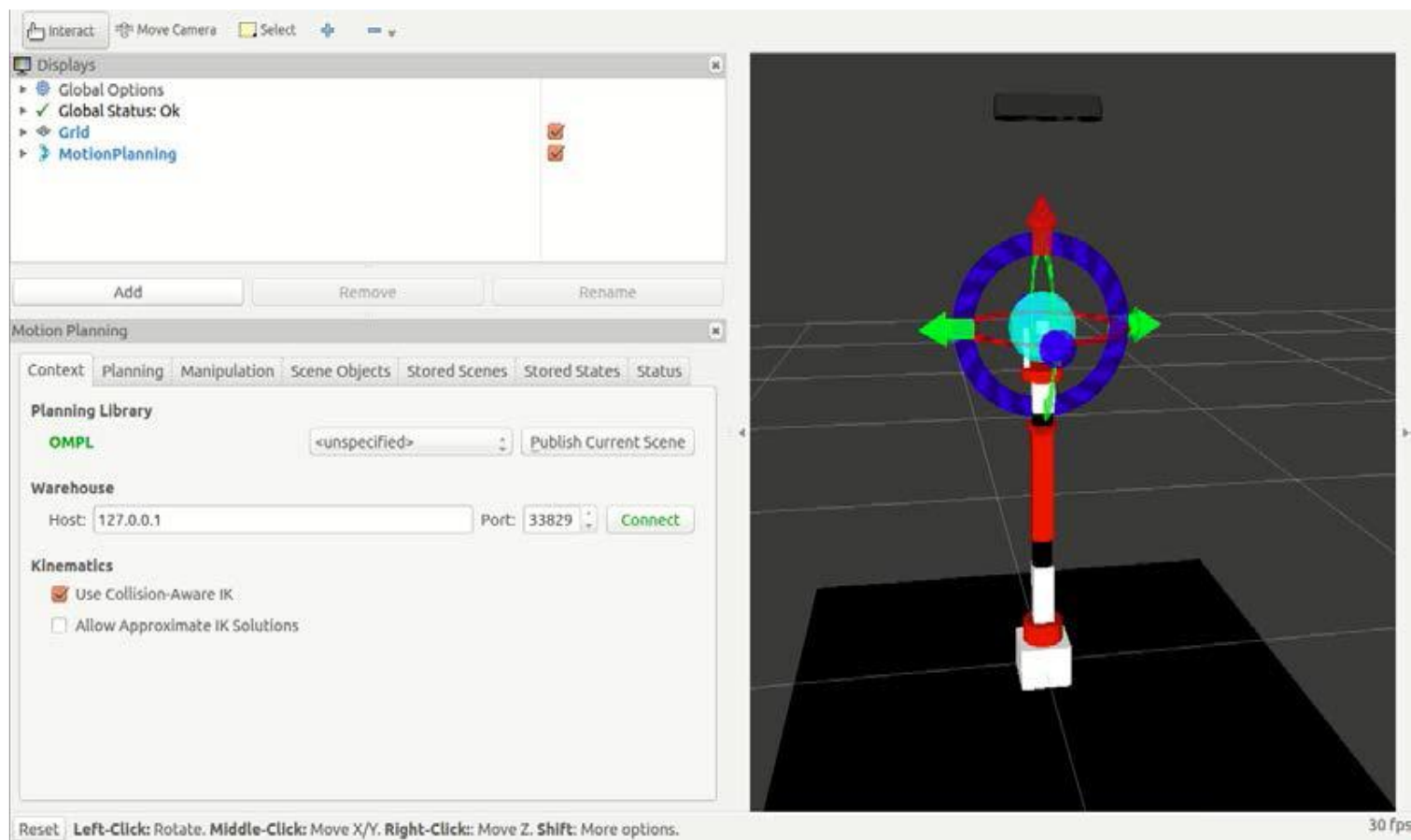
## 生成配置文件



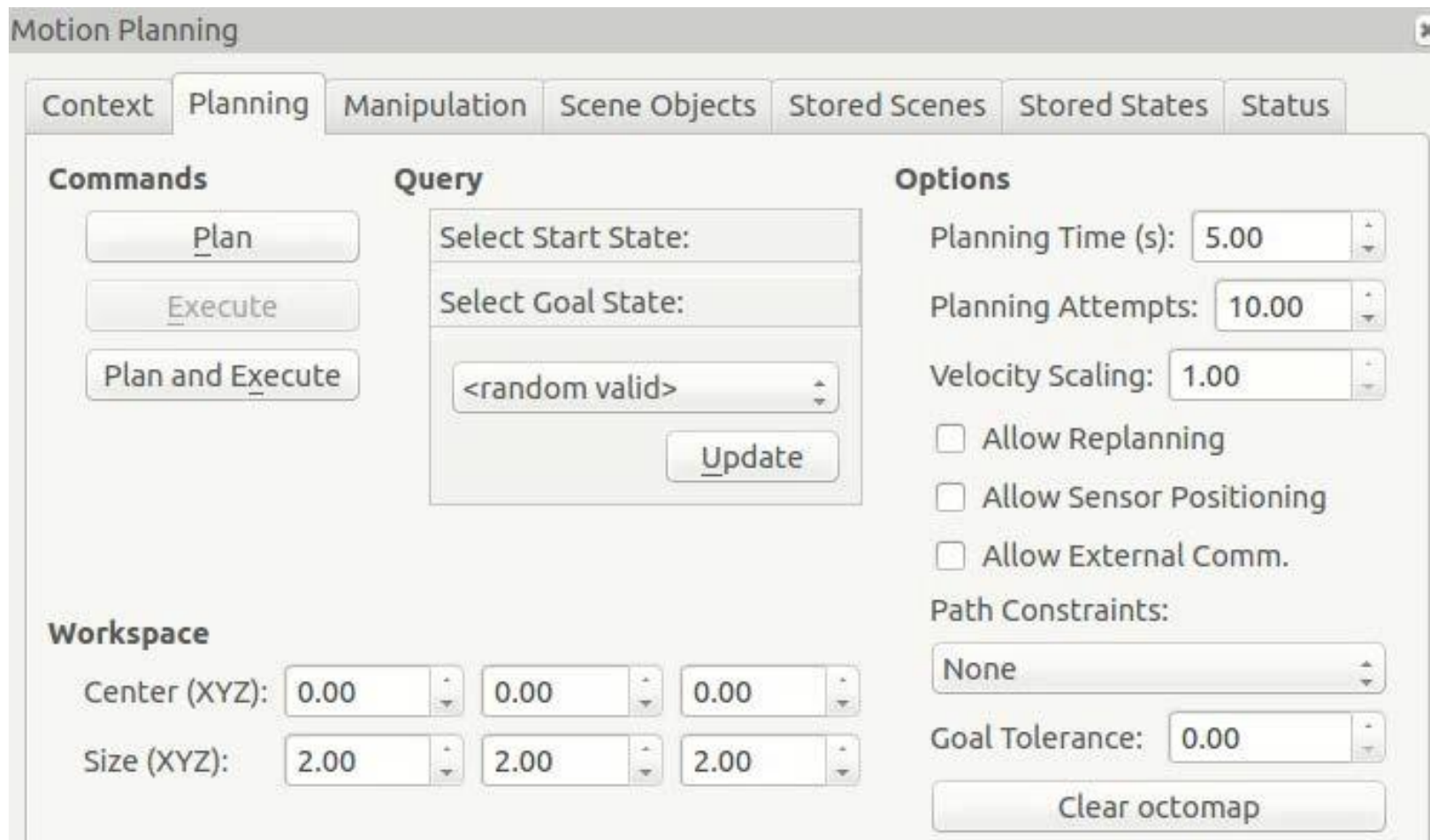
# RVIZ中Moveit规划



\$ roslaunch \*\*\* demo.launch



# RVIZ中Moveit规划



谢谢！

