World MoveIt! Day 2019 China Developer Workshop

Hands-on Scripts

Jan. 2019

Prerequisite (in dock image): (5 mins)

The installation and environment configuration in this section are already included in the docker image we provided. You can skip this section.

- Ubuntu 16.04
- ROS Kinetic desktop full http://wiki.ros.org/kinetic/Installation/Ubuntu
- MoveIt http://moveit.ros.org/install/

sudo apt install ros-kinetic-moveit ros-kinetic-moveit-resources ros-kinetic-moveit-visual-tools ros-kinetic-panda-moveit-config ros-kinetic-geometric-shapes ros-kinetic-tf2-geometry-msgs

Install necessary debian packages:

```
sudo apt install python-rosinstall python-rosinstall-generator python-wstool build-essential
```

```
sudo apt install python-wstool python-catkin-tools clang-format-3.8
```

```
sudo apt install ros-kinetic-universal-robot
```

```
sudo apt install ros-kinetic-ur-description
```

Install handson source code:

```
echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc
```

```
mkdir -p ~/ws_handson/src
```

```
cd ~/ws_handson/src
```

```
git clone https://github.com/RoboticsYY/moveit_handson.git
```

```
git clone https://github.com/RoboticsYY/moveit_core_handson.git
```

```
git clone https://github.com/RoboticsYY/moveit_ros_handson.git
```

```
cd ..
```

```
catkin_make -DCMAKE_BUILD_TYPE=Release
```

More information:

Onsite technical support: OTC Robotics Engineering Team

Online version of this handson scripts for quick following:

https://roboticsyy.github.io/handson_tutorial/index.html

0. Launch dock image

Some commands need to be run by sudoers on the laptop, the password is: intel.

Please follow the command below to install and update docker:

```
sudo apt update && sudo apt install -y wget
```

```
mkdir -p ~/code/ && cd ~/code
```

```
wget https://raw.githubusercontent.com/RoboticsYY/moveit_handson/master/docker/install_docker.sh
```

```
wget https://raw.githubusercontent.com/RoboticsYY/moveit_handson/master/docker/setup_docker_display.sh
```

```
chmod +x install_docker.sh
```

```
./install_docker.sh
```

Add your user to the docker group.

```
sudo usermod -aG docker $USER
```

Log out and log back in so that your group membership is re-evaluated. On a desktop Linux environment such as X Windows, log out of your session completely and then log back in.

```
gnome-session-quit
```

If you have a laptop we provided, you can skip this step. If you have a USB stick we provided, the stick contains the handson docker image `moveit_handson_demo.tar.tgz`. Copy the image to the local disk at first, then please follow the description below to decompress and load the image:

```
sudo apt update && sudo apt install -y tar
```

```
tar -zxvf moveit_handson_demo.tar.tgz
```

```
docker load < moveit_handson_demo.tar
```

Please refer below command to verify the docker image created successfully on the disk. Run in shell:

```
docker images
```

It should show the following information:

```
REPOSITORY       TAG                   IMAGE ID        CREATED          SIZE
intel/kinetic    moveit_handson_demo   6643cc4db2e5    5 hours ago      3.47GB
```

Run following commands in one host terminal to set the operating environment first so that x window can pop up within the docker container:

```
cd ~/code
```

```
chmod +x setup_docker_display.sh
```

```
./setup_docker_display.sh
```

```
docker run -t -i --rm -v /tmp/.X11-unix:/tmp/.X11-unix:rw -v /tmp/.docker.xauth:/tmp/.docker.xauth:rw -e XAUTHORITY=/tmp/.docker.xauth -e DISPLAY --name moveit_handson intel/kinetic:moveit_handson_demo bash
```

Note: You should not exit the docker container of this terminal during the whole handson process. Otherwise, your operations in the docker container cannot be saved.

If you want more information on how to use docker, please see the article here for more information: https://docs.docker.com/install/

Note: If you don’t have a USB stick, you can install the docker image with the following command:

```
docker pull congliu0913/moveit_handson_demo
```

1. Getting Started:

1.1 Setup the handson code: (5 mins)

Run from a second host terminal to load the docker container:

```
docker exec -t -i moveit_handson bash
```

Compile the handson code:

```
catkin_make -DCMAKE_BUILD_TYPE=Release
```

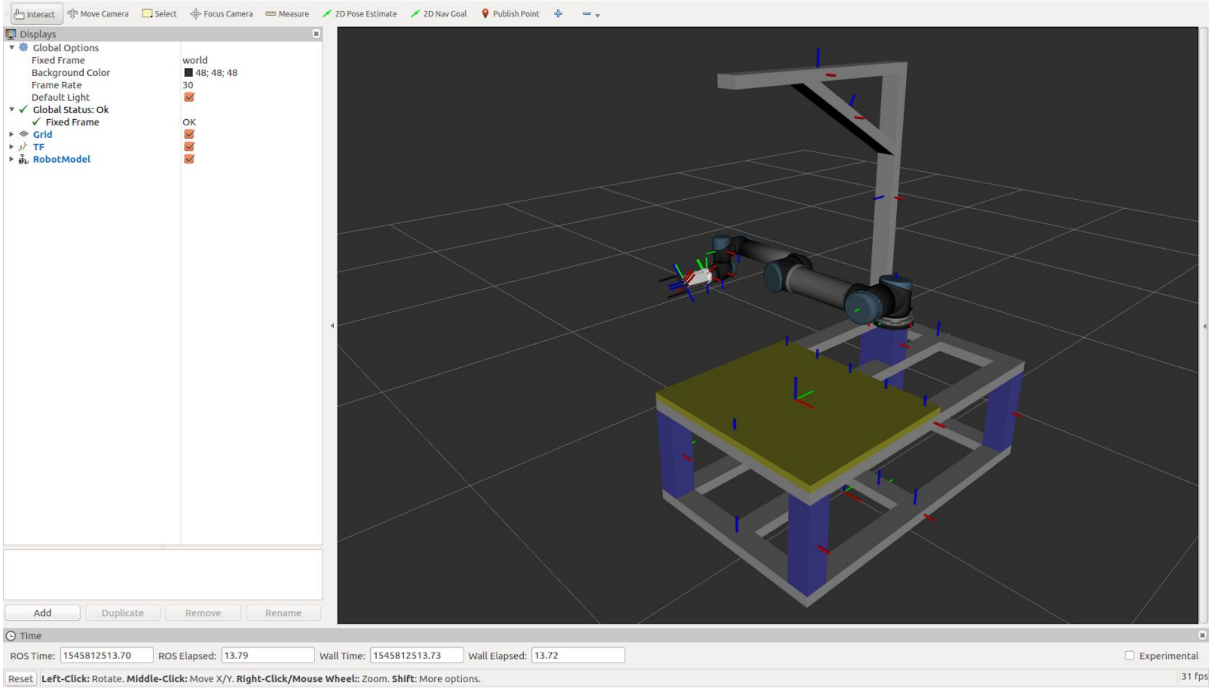
Source the catkin environment:

```
source devel/setup.bash
```

You can check the installation by run:

```
roslaunch handson_description visualize_ur5.launch
```

If everything works fine, you can see the following screen. Then Ctrl-C to exit the roslaunch and finish the check.



Global Options
Fixed Frame world
Background Color ■ 48; 48; 48
Frame Rate 30
Default Light ☑
✓ Global Status: Ok
✓ Fixed Frame OK
Grid ☑
TF ☑
RobotModel ☑

ROS Time: 1545812513.70 ROS Elapsed: 13.79 Wall Time: 1545812513.73 Wall Elapsed: 13.72 □ Experimental

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift:** More options. 31 fps

2. MoveIt! Configure (30 mins)

MoveIt! config package is used to bring up the MoveIt! motion planning, perception and pick place pipeline. If you have the `ur5_hitbot_ilc_platform_moveit_config` package already git-cloned from the **Getting Started** section, don't change anything in this package. This is the MoveIt! config package we created previously, you can refer to this package when you meet error in the rest of the handson. Next, we will show you how to create the same config package.

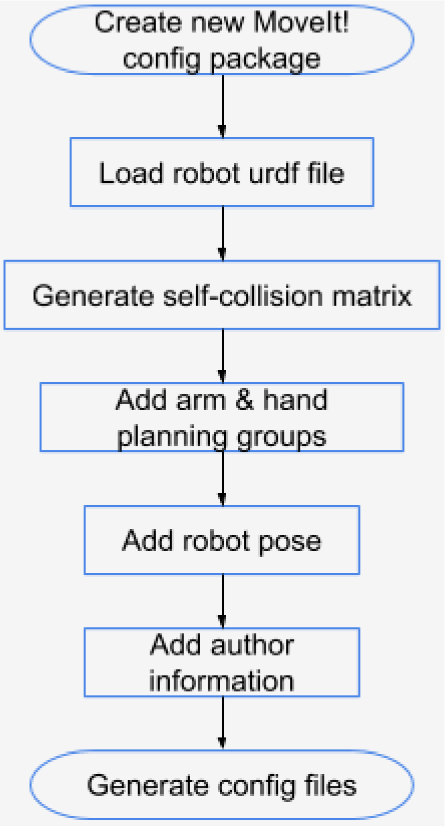
2.1 Start MoveIt! Setup Assistant (2 mins)

```
docker exec -t -i moveit_handson bash
```

```
source devel/setup.bash
```

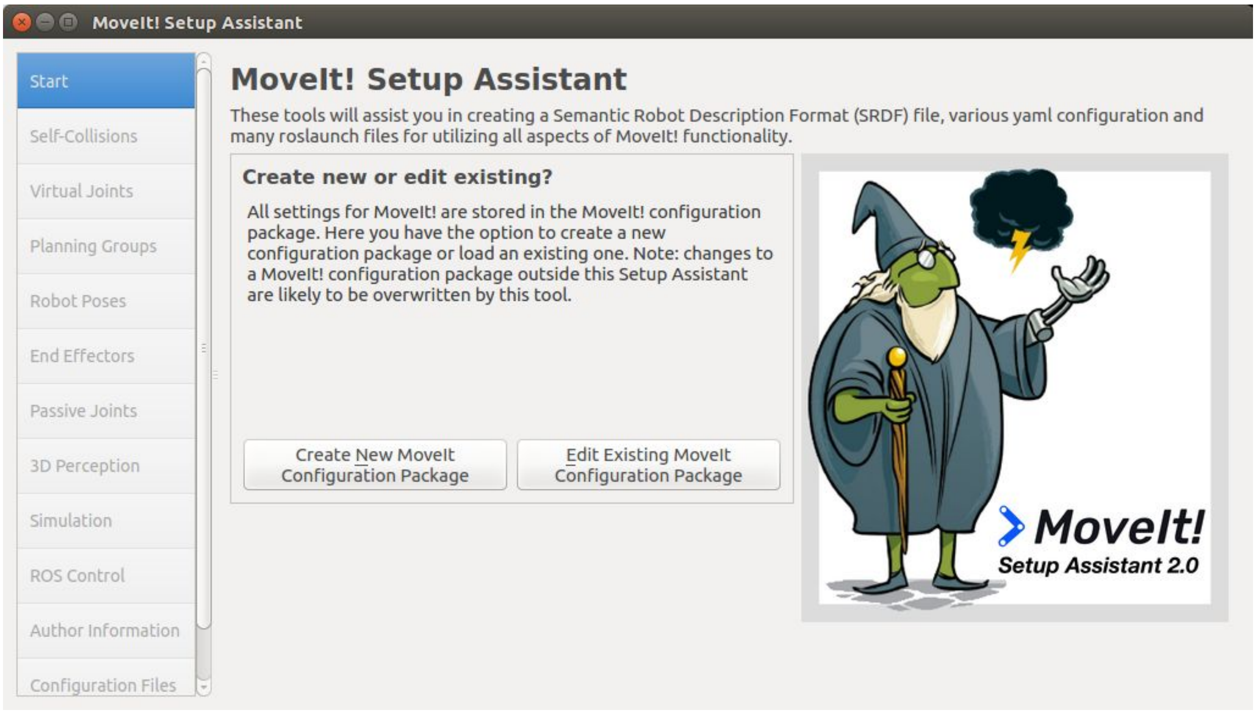
```
roslaunch moveit_setup_assistant setup_assistant.launch
```

The operation flow in this section will be:


Create new MoveIt! config package
Load robot urdf file
Generate self-collision matrix
Add arm & hand planning groups
Add robot pose
Add author information
Generate config files

2.2 Create new MoveIt! config package (1 mins)

Click on the **Create New MoveIt! Configuration Package** button on the **Start** screen:



MoveIt! Setup Assistant

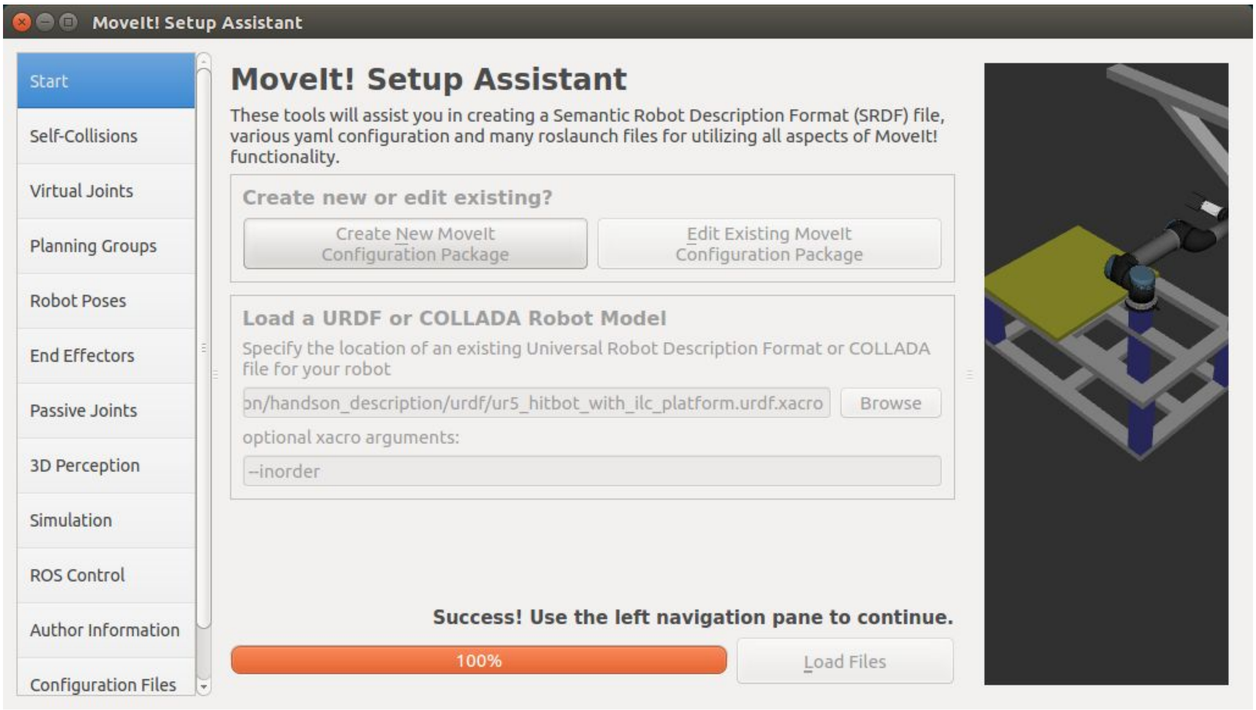
These tools will assist you in creating a Semantic Robot Description Format (SRDF) file, various yaml configuration and many roslaunch files for utilizing all aspects of MoveIt! functionality.

Create new or edit existing?

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.

2.3 Load urdf file (1 mins)

Click on the **browse** button and navigate to the **ur5_hitbot_with_ilc_platform.urdf.xacro** file in the **handson_description** package. (this file gets installed in **/root/ws_handson/src/moveit_handson/handson_description/urdf/ur5_hitbot_with_ilc_platform .urdf.xacro**) Choose that file and then click **Load Files**. The Setup Assistant will load the files (this might take a few seconds) and present you with this 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?

Load a URDF or COLLADA Robot Model

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

on/handson_description/urdf/ur5_hitbot_with_ilc_platform.urdf.xacro

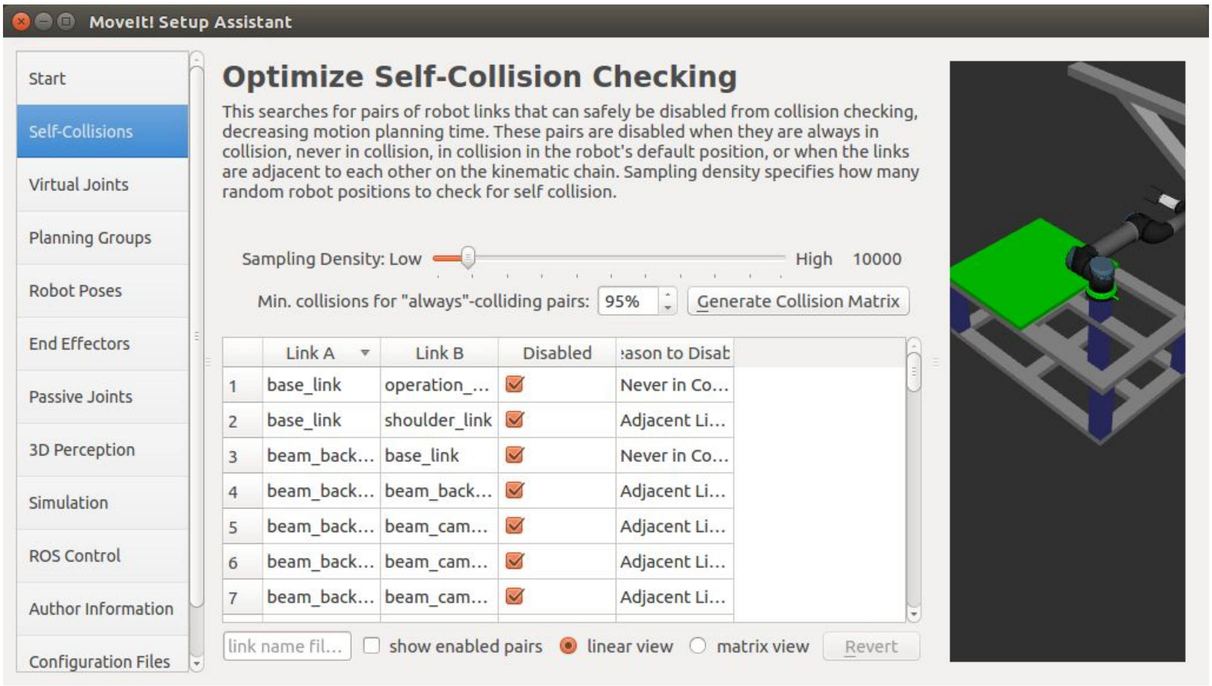
optional xacro arguments:

--inorder

Success! Use the left navigation pane to continue.

2.4 Generate self-collision matrix (2 mins)

Click on the **Self-Collisions** pane selector on the left-hand side and click on the **Generate Collision Matrix** button. The Setup Assistant will work for a few seconds before presenting you the results of its computation in the main table:



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 random robot positions to check for self collision.

Sampling Density: Low High 10000

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

	Link A	Link B	Disabled	ason to Disab
1	base_link	operation_...	☑	Never in Co...
2	base_link	shoulder_link	☑	Adjacent Li...
3	beam_back...	base_link	☑	Never in Co...
4	beam_back...	beam_back...	☑	Adjacent Li...
5	beam_back...	beam_cam...	☑	Adjacent Li...
6	beam_back...	beam_cam...	☑	Adjacent Li...
7	beam_back...	beam_cam...	☑	Adjacent Li...

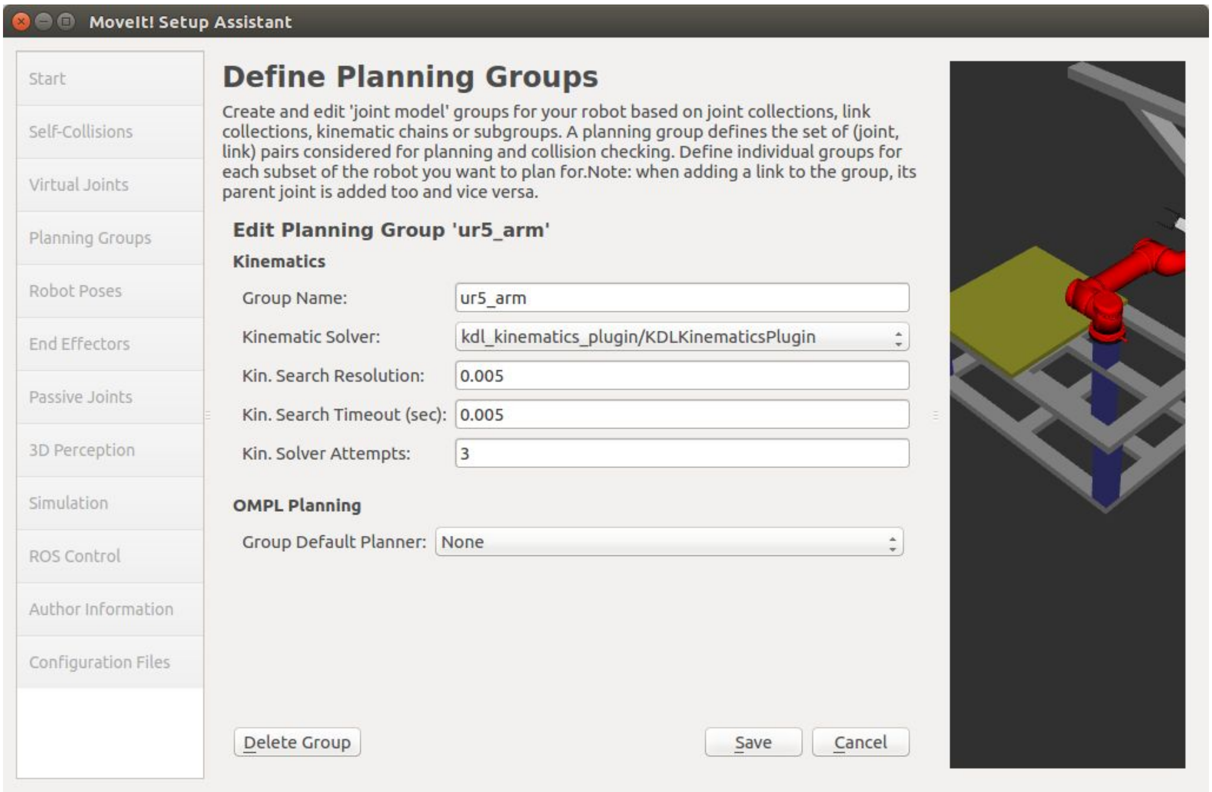
□ show enabled pairs ● linear view ○ matrix view

2.5 Virtual joints (1 mins)

Virtual joints are used primarily to attach the robot to the world. Since the UR5 robot is already attached to the platform and the platform is attached to the world in the loaded urdf file, there is no need to add virtual joints.

2.6 Add arm & hand planning groups (8 mins)

- Click on the **Planning-Groups** panel selector
- Click on **Add Group** button to add arm group:
- We will first add UR5 arm as a planning group:
  - Enter **Group Name** as “ur5_arm”
  - Choose **kdl_kinematics_plugin/KDLKinematicsPlugin** as the kinematics solver.
  - Let **Kin. Search Resolution** and **Kin. Search Timeout** stay at their default values.



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.Note: when adding a link to the group, its parent joint is added too and vice versa.

Edit Planning Group 'ur5_arm'

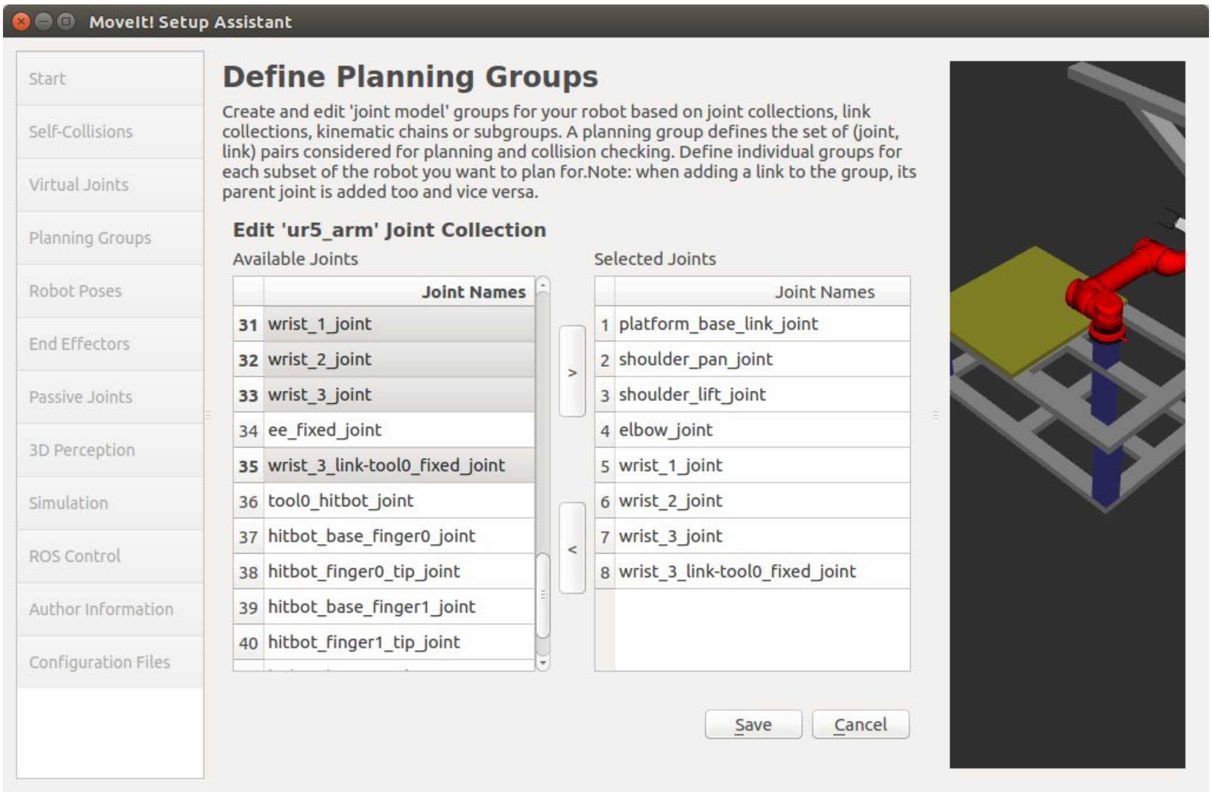
Kinematics

Group Name:	ur5_arm
Kinematic Solver:	kdl_kinematics_plugin/KDLKinematicsPlugin
Kin. Search Resolution:	0.005
Kin. Search Timeout (sec):	0.005
Kin. Solver Attempts:	3

OMPL Planning

Group Default Planner: None

- Now, click on the **Add Joints** button:
  - Choose "platform_base_link_joint", "shoulder_pan_joint", "shoulder_lift_joint", "elbow_joint", "wrist_1_joint", "wrist_2_joint", "wrist_3_joint" and "wrist_3_link-tool0_fixed_joint" from the left **Available Joints** list. Add them to the right **Selected Joints** list.
- Click **Save** button to save the selected group:



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.Note: when adding a link to the group, its parent joint is added too and vice versa.

Edit 'ur5_arm' Joint Collection

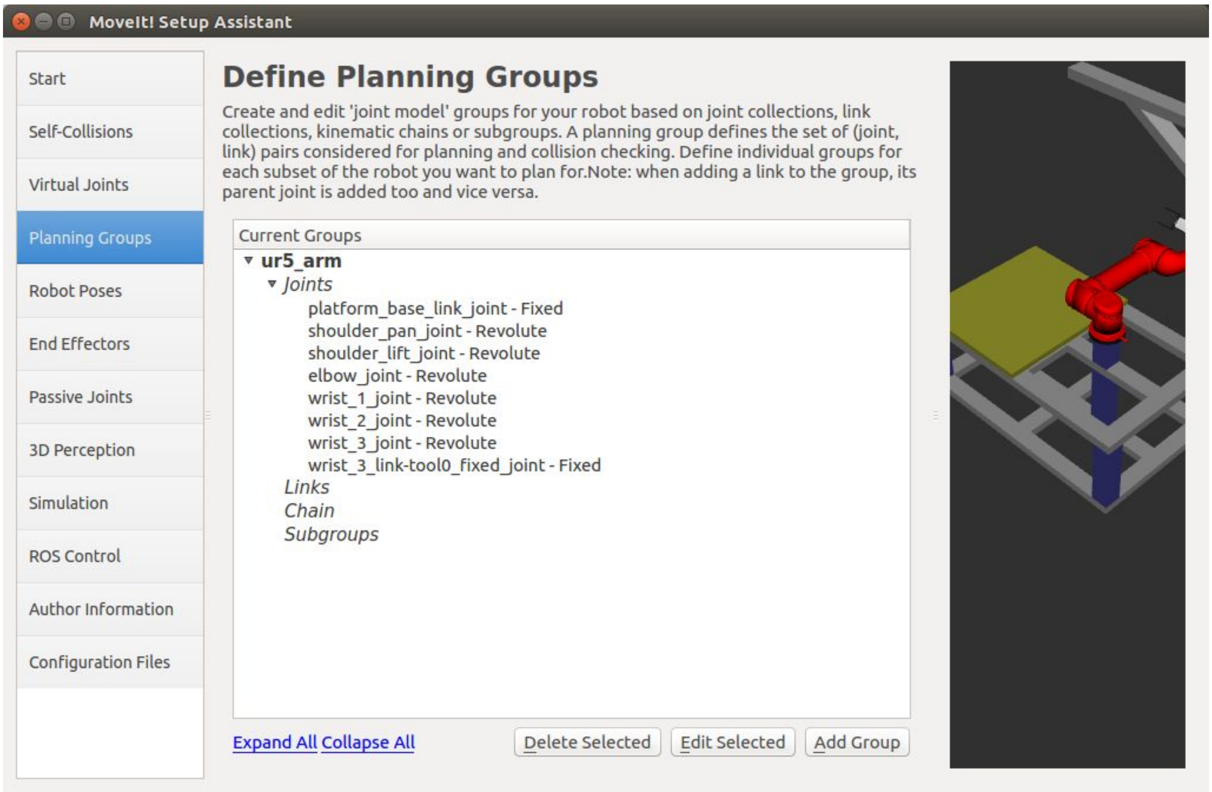
Available Joints

	Joint Names
31	wrist_1_joint
32	wrist_2_joint
33	wrist_3_joint
34	ee_fixed_joint
35	wrist_3_link-tool0_fixed_joint
36	tool0_hitbot_joint
37	hitbot_base_finger0_joint
38	hitbot_finger0_tip_joint
39	hitbot_base_finger1_joint
40	hitbot_finger1_tip_joint

Selected Joints

	Joint Names
1	platform_base_link_joint
2	shoulder_pan_joint
3	shoulder_lift_joint
4	elbow_joint
5	wrist_1_joint
6	wrist_2_joint
7	wrist_3_joint
8	wrist_3_link-tool0_fixed_joint

- Now, the screen should be like this:

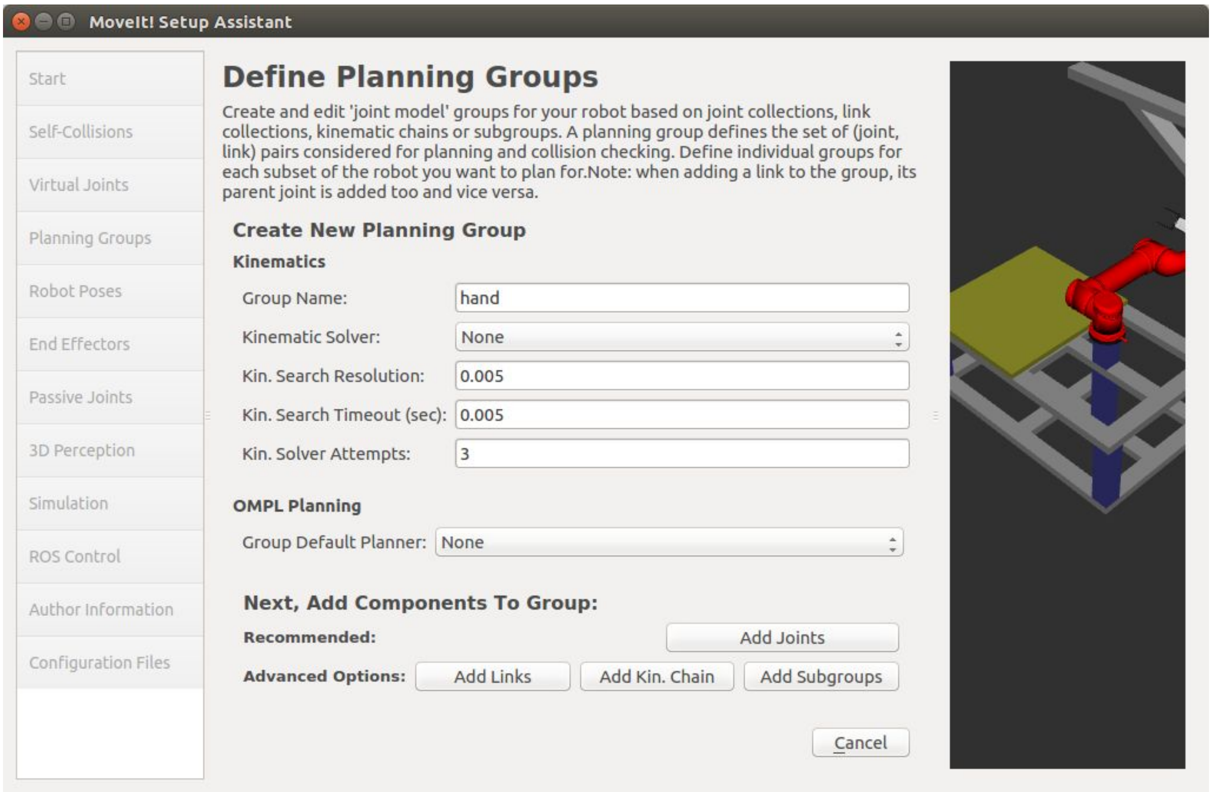


Define Planning Groups

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.Note: when adding a link to the group, its parent joint is added too and vice versa.

Current Groups
- **ur5_arm**
  - *Joints*
    - platform_base_link_joint - Fixed
    - shoulder_pan_joint - Revolute
    - shoulder_lift_joint - Revolute
    - elbow_joint - Revolute
    - wrist_1_joint - Revolute
    - wrist_2_joint - Revolute
    - wrist_3_joint - Revolute
    - wrist_3_link-tool0_fixed_joint - Fixed
  - *Links*
  - *Chain*
  - *Subgroups*

- Click **Add Group** button again to add the end-effector group. NOTE that you will do this using a different procedure than adding the arm:
  - Enter **Group Name** as “hand”
  - Let **Kin. Search Resolution** and **Kin. Search Timeout** stay at their default values.



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.Note: when adding a link to the group, its parent joint is added too and vice versa.

Create New Planning Group

Kinematics

Group Name: hand
Kinematic Solver: None
Kin. Search Resolution: 0.005
Kin. Search Timeout (sec): 0.005
Kin. Solver Attempts: 3

OMPL Planning

Group Default Planner: None

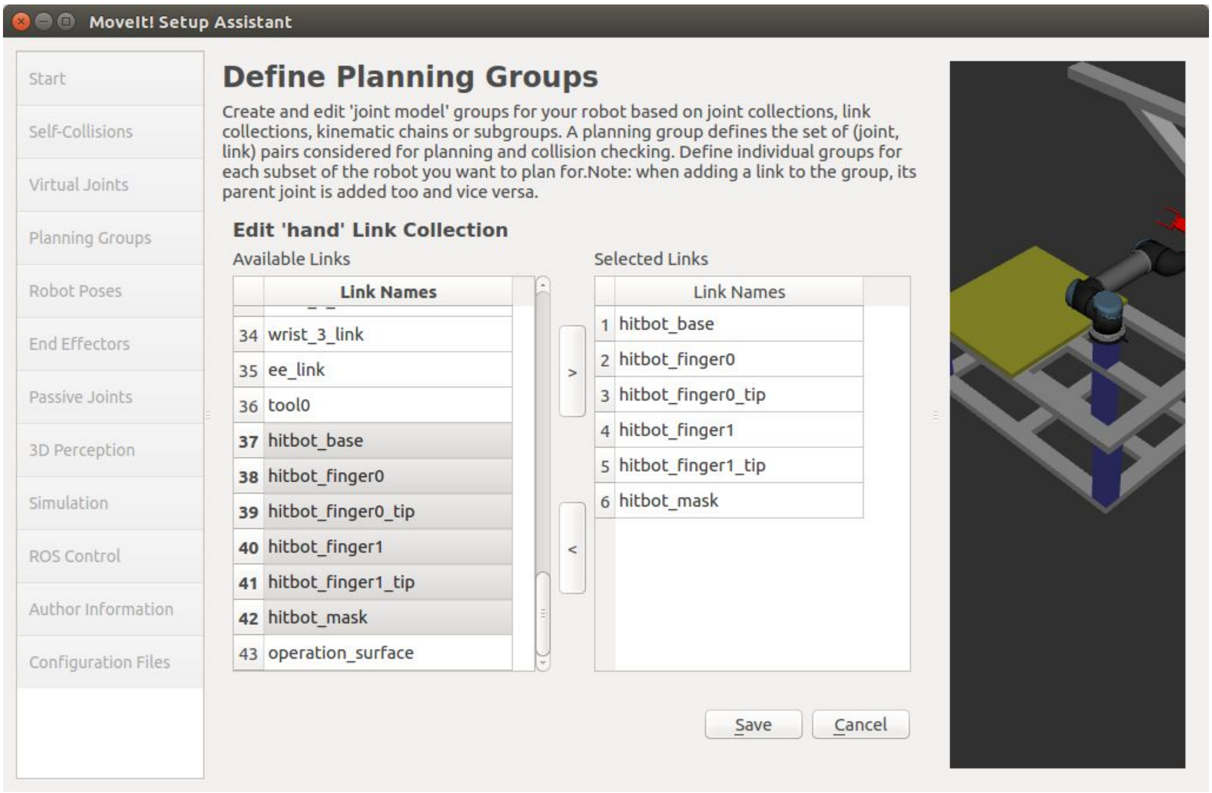
Next, Add Components To Group:

Recommended:

Advanced Options:

  - Click on the **Add Links** button.
  - Choose “hitbot_base”, “hitbot_mask”, “hitbot_finger0”, “hitbot_finger0_tip”, “hitbot_finger1” and “hitbot_finger1_tip”, and add them to the list of **Selected Links** on the right hand side.

- Click **Save**



MoveIt! Setup Assistant

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.Note: when adding a link to the group, its parent joint is added too and vice versa.

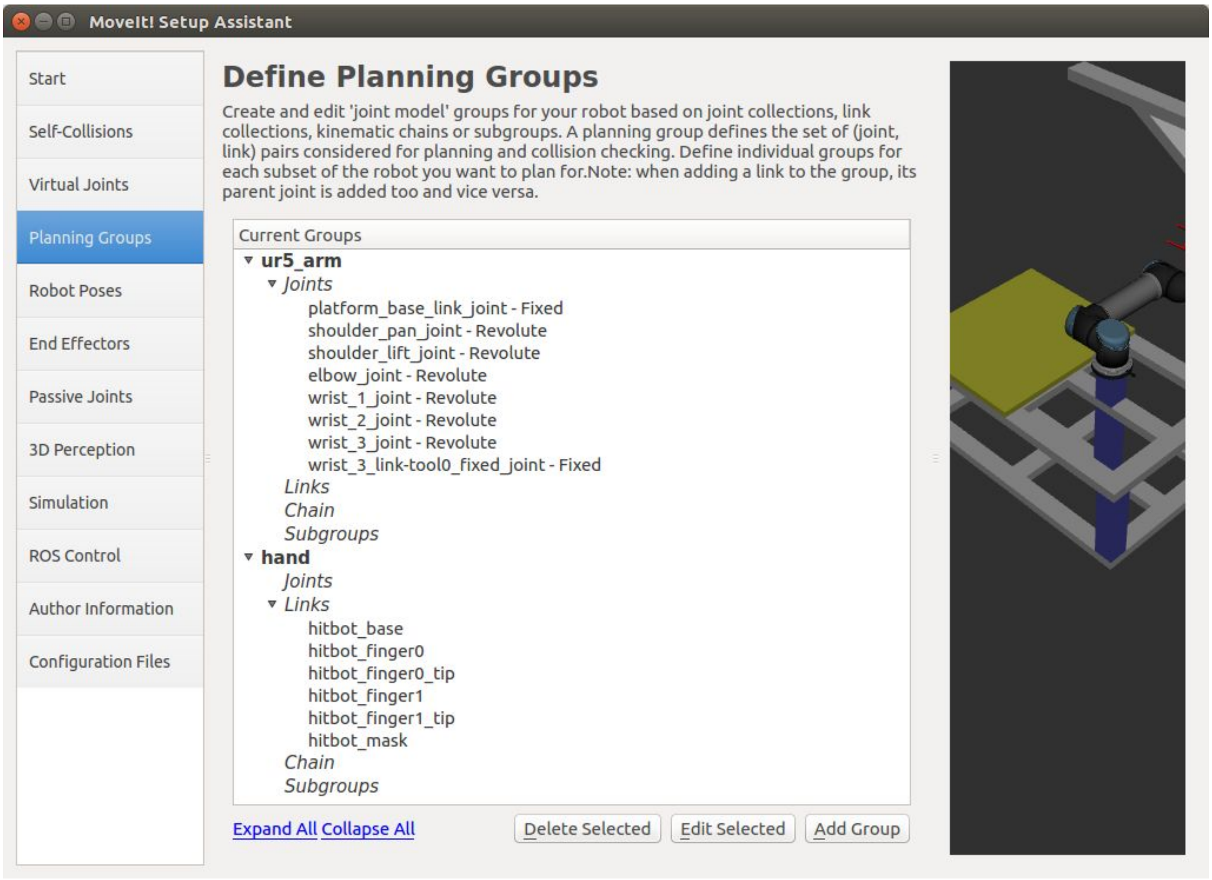
Edit 'hand' Link Collection

Available Links

	Link Names
34	wrist_3_link
35	ee_link
36	tool0
37	hitbot_base
38	hitbot_finger0
39	hitbot_finger0_tip
40	hitbot_finger1
41	hitbot_finger1_tip
42	hitbot_mask
43	operation_surface

Selected Links

	Link Names
1	hitbot_base
2	hitbot_finger0
3	hitbot_finger0_tip
4	hitbot_finger1
5	hitbot_finger1_tip
6	hitbot_mask



MoveIt! Setup Assistant

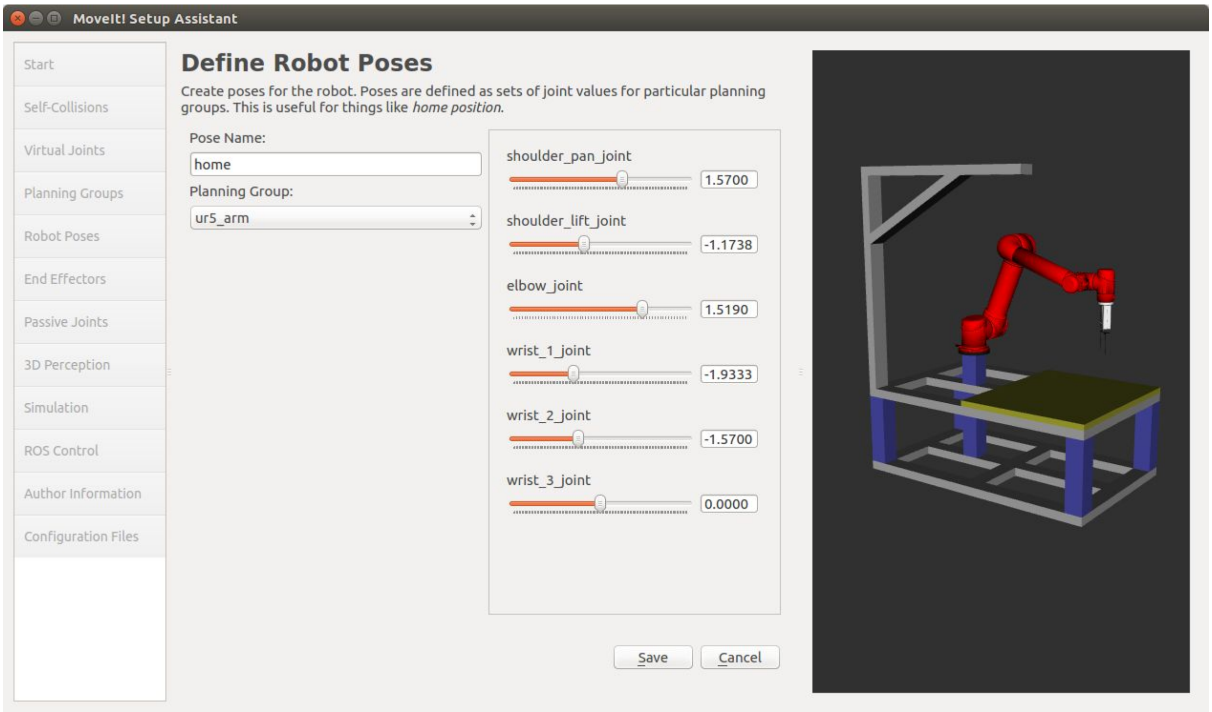
Define Planning Groups

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.Note: when adding a link to the group, its parent joint is added too and vice versa.

Current Groups

```
▾ ur5_arm
  ▾ Joints
      platform_base_link_joint - Fixed
      shoulder_pan_joint - Revolute
      shoulder_lift_joint - Revolute
      elbow_joint - Revolute
      wrist_1_joint - Revolute
      wrist_2_joint - Revolute
      wrist_3_joint - Revolute
      wrist_3_link-tool0_fixed_joint - Fixed
    Links
    Chain
    Subgroups
▾ hand
    Joints
  ▾ Links
      hitbot_base
      hitbot_finger0
      hitbot_finger0_tip
      hitbot_finger1
      hitbot_finger1_tip
      hitbot_mask
    Chain
    Subgroups
```

2.7 Add robot poses (5 mins)



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.*

Pose Name: home

Planning Group: ur5_arm

shoulder_pan_joint 1.5700

shoulder_lift_joint -1.1738

elbow_joint 1.5190

wrist_1_joint -1.9333

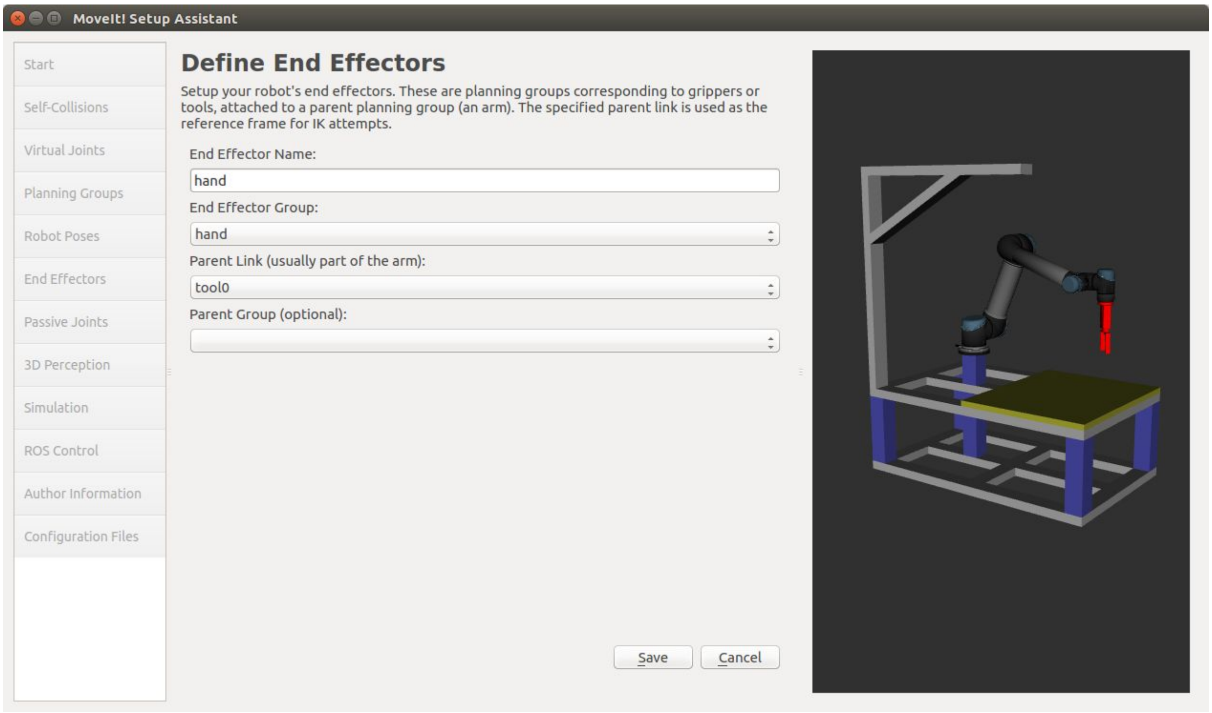
wrist_2_joint -1.5700

wrist_3_joint 0.0000

- Click on the **Robot Poses** panel
- Click **Add Pose** button
- Set “home” as **Pose Name**
- Choose “ur5_arm” as **Planning Group**
  - Set “shoulder_pan_joint” as “1.57”
  - Set “shoulder_lift_joint” as “-1.1738”
  - Set “elbow_joint” as “1.5190”
  - Set “wrist_1_joint” as “-1.9333”
  - Set “wrist_2_joint” as “-1.57”
  - Set “wrist_3_joint” as “0.0”
- Click **Save** button to save the pose

2.8 Label end-effectors (5 mins)

- Click on the **End Effectors** panel.
- Click **Add End Effector**.
- Choose “hand” as the **End Effector Name** for the gripper.
- Select “hand” as the **End Effector Group**.
- Select “tool0” as the **Parent Link** for this end-effector.
- Leave **Parent Group** blank.
- Click **Save.**



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: hand

End Effector Group: hand

Parent Link (usually part of the arm): tool0

Parent Group (optional):

Note: The screens of **Passive Joints**, **3D Perception**, **Simulation** and **ROS Control** can be skipped, they are not necessary right now.

The passive joints are the unactuated joints of the robot, since UR5 doesn’t have such kind of joints, so the **Passive Joints** panel will be skipped.

The **3D Perception** is used to config the parameters of octomap by using a 3D sensor, such as RGBD cameras, stereo cameras or laser scanners. In this handson, we will skip this.

The **Simulation** will add necessary tags in the urdf file, so that the physics of the robot will be simulated in Gazebo. Since we have already added these tags manually, so this step can be skipped.

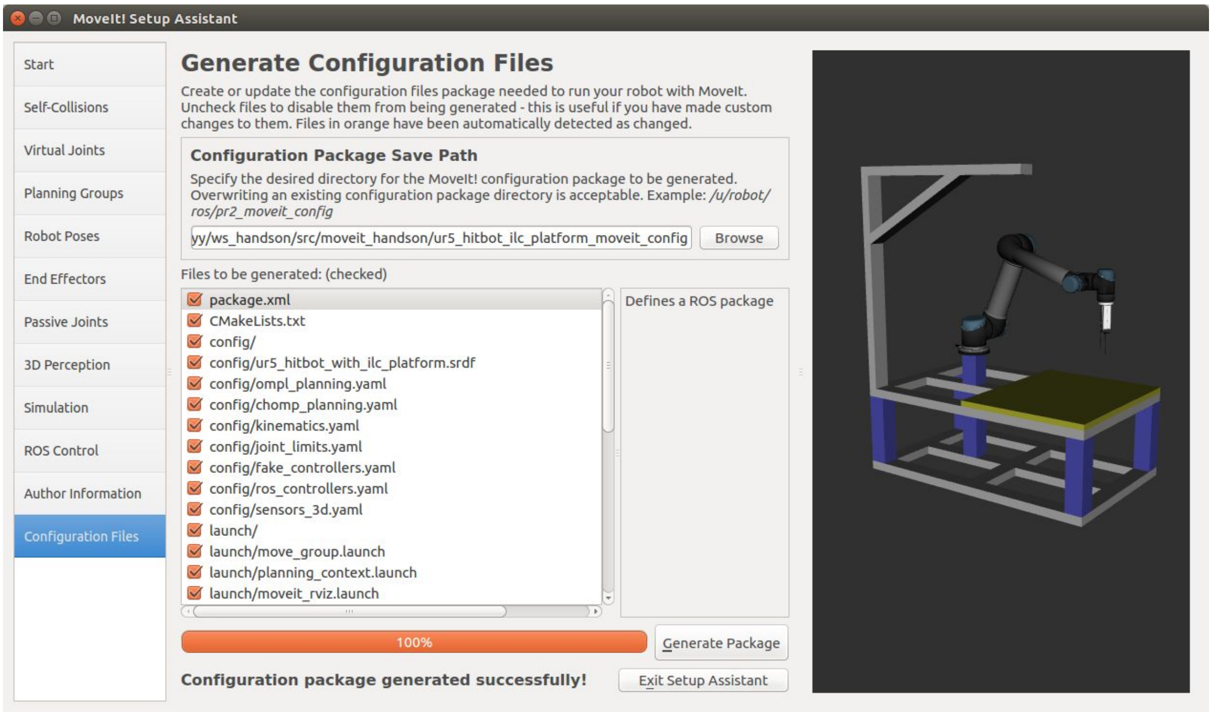
Since the handson will be implemented mainly in a simulation environment. You can also skip the **ROS Control** panel, which is used to config the controller parameters for real robot execution.

2.9 Author information (1 mins)

Click on the **Author Information** panel. Enter your name and email address.

2.10 Generate configuration files (1 mins)

- Click on the Configuration Files panel.
- In the **Configuration Package Save Path**, browse to the file location `/root/ws_handson/src/moveit_handson`, and input `/handson_moveit_config` as your package name.
- Click on the **Generate Package button**.
- Click **Exit Setup Assistant**. Now, you can find your moveit config package in `/root/ws_handson/src/moveit_handson/handson_moveit_config`.



MoveIt! Setup Assistant

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/pr2_moveit_config*

yy/ws_handson/src/moveit_handson/ur5_hitbot_ilc_platform_moveit_config [Browse]

Files to be generated: (checked)

- ☑ package.xml
- ☑ CMakeLists.txt
- ☑ config/
- ☑ config/ur5_hitbot_with_ilc_platform.srdf
- ☑ config/ompl_planning.yaml
- ☑ config/chomp_planning.yaml
- ☑ config/kinematics.yaml
- ☑ config/joint_limits.yaml
- ☑ config/fake_controllers.yaml
- ☑ config/ros_controllers.yaml
- ☑ config/sensors_3d.yaml
- ☑ launch/
- ☑ launch/move_group.launch
- ☑ launch/planning_context.launch
- ☑ launch/moveit_rviz.launch

Defines a ROS package

100%

Configuration package generated successfully!

2.11 Set initial pose to fake_controllers.yaml (1 mins)

Set “home” pose as the initial pose for the simulation:

Open `handson_moveit_config/config/fake_controllers.yaml` with any editor, add the following lines to the end of the file, save and close:

```
initial:
 - group: ur5_arm
   pose:  home
```

2.12 Adjust moveit.rviz (2 mins)

Open `handson_moveit_config/launch/moveit.rviz`, add `RvizVisualToolsGui` to the subitems of `Panels`, so that it would look like:

```
Panels:
  - Class: rviz/Displays
    Help Height: 84
    Name: Displays
    Property Tree Widget:
      Expanded: ~
      Splitter Ratio: 0.742560029
    Tree Height: 330
  - Class: rviz/Help
    Name: Help
  - Class: rviz/Views
    Expanded:
      - /Current View1
    Name: Views
    Splitter Ratio: 0.5
  - Class: rviz_visual_tools/RvizVisualToolsGui
    Name: RvizVisualToolsGui
...
```

Still in the `handson_moveit_config/launch/moveit.rviz` file, add `MarkerArray` to the `Displays` panel, so it would look like:

```
Visualization Manager:
  Class: ""
  Displays:
    - Class: rviz/MarkerArray
      Enabled: true
      Marker Topic: /rviz_visual_tools
      Name: MarkerArray
      Namespaces:
        Text: true
      Queue Size: 100
      Value: true
    - Alpha: 0.5
      Cell Size: 1
      Class: rviz/Grid
      Color: 160; 160; 164
      Enabled: true
      Line Style:
        Line Width: 0.03
        Value: Lines
      Name: Grid
      Normal Cell Count: 0
      Offset:
        X: 0
        Y: 0
        Z: 0
...
```

3. Motion Planning and Execution (10 mins)

Bring up MoveIt! motion planning pipeline in one shell. Remember to source the workspace before executing the command:

```
docker exec -t -i moveit_handson bash
```

```
source devel/setup.bash
```

```
roslaunch handson_moveit_config demo.launch
```

Note: this command is executed from the config package just created, if you meet error in this section, you can replace the above command by the following command to see the result:

```
roslaunch ur5_hitbot_ilc_platform_moveit_config demo.launch
```

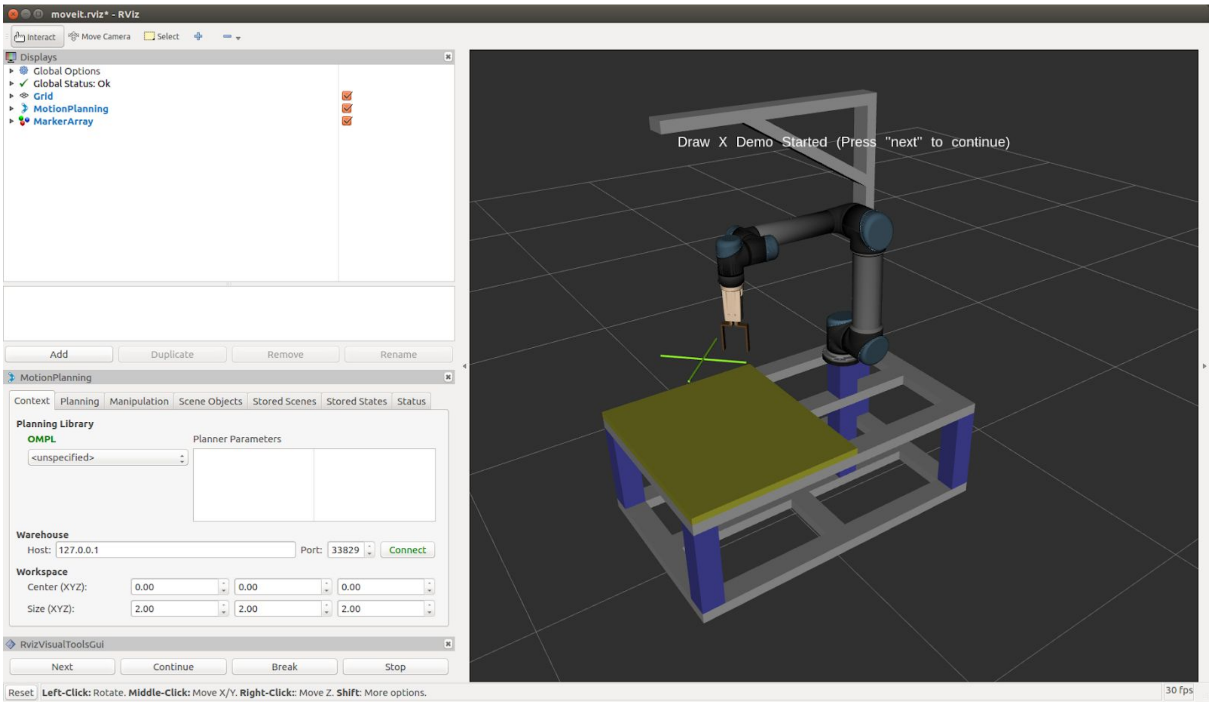
In another shell, run the demo that makes UR5 end-effector to draw letter “X”:

```
docker exec -t -i moveit_handson bash
```

```
source devel/setup.bash
```

```
roslaunch handson_example draw_x.launch
```

Press the “Next” button on the RvizVisualToolsGui to start the demo



OMPL <unspecified>

Warehouse
Host: 127.0.0.1 Port: 33829

Workspace
Center (XYZ): 0.00 0.00 0.00
Size (XYZ): 2.00 2.00 2.00

Draw X Demo Started (Press "next" to continue)

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click::** Move Z. **Shift**: More options.

4. Pick and Place (10 mins)

Bring up MoveIt! motion planning pipeline in one shell. Remember to source the workspace before executing the command:

```
docker exec -t -i moveit_handson bash
```

```
source devel/setup.bash
```

```
roslaunch handson_moveit_config demo.launch
```

Note: this command is executed from the config package just created, if you meet error in this section, you can replace the above command by the following command to see the result:

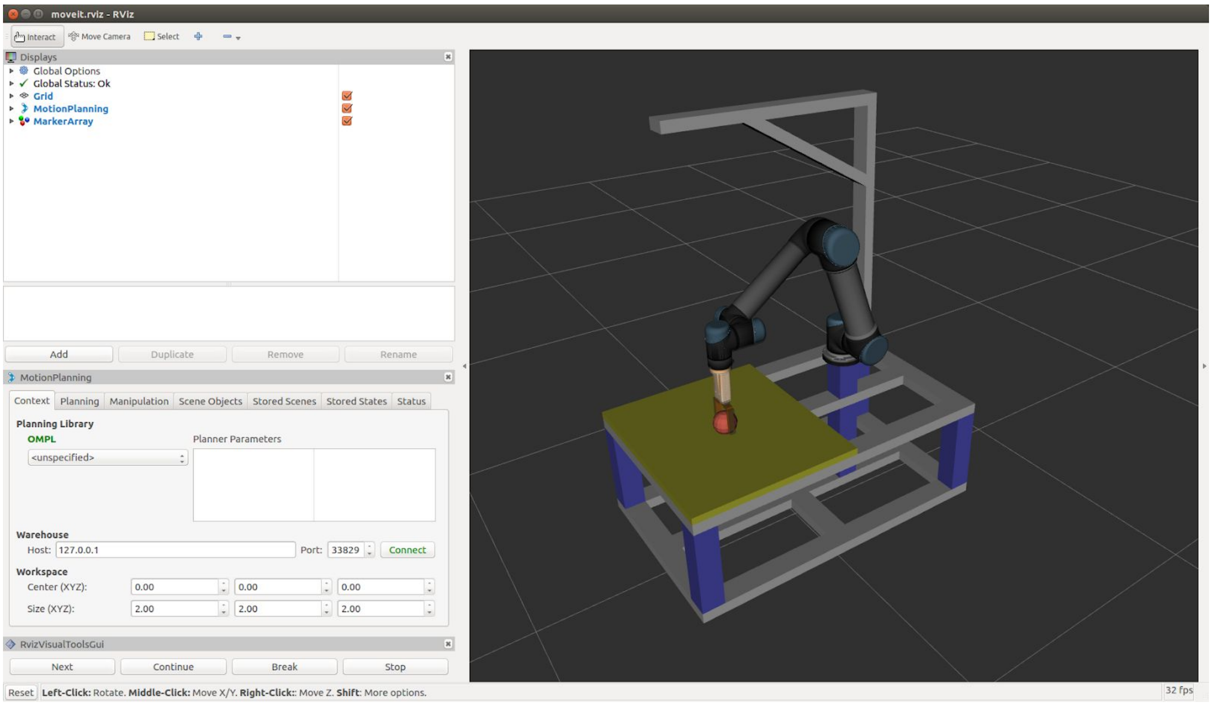
```
roslaunch ur5_hitbot_ilc_platform_moveit_config demo.launch
```

Run the pick and place demo, in which the UR5 robot will pick a ball with hitbot gripper from the platform surface and place it at another location:

```
docker exec -t -i moveit_handson bash
```

```
source devel/setup.bash
```

```
roslaunch handson_example pick_place.launch
```



Host: 127.0.0.1 Port: 33829

Center (XYZ): 0.00 0.00 0.00

Size (XYZ): 2.00 2.00 2.00

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click::** Move Z. **Shift:** More options.