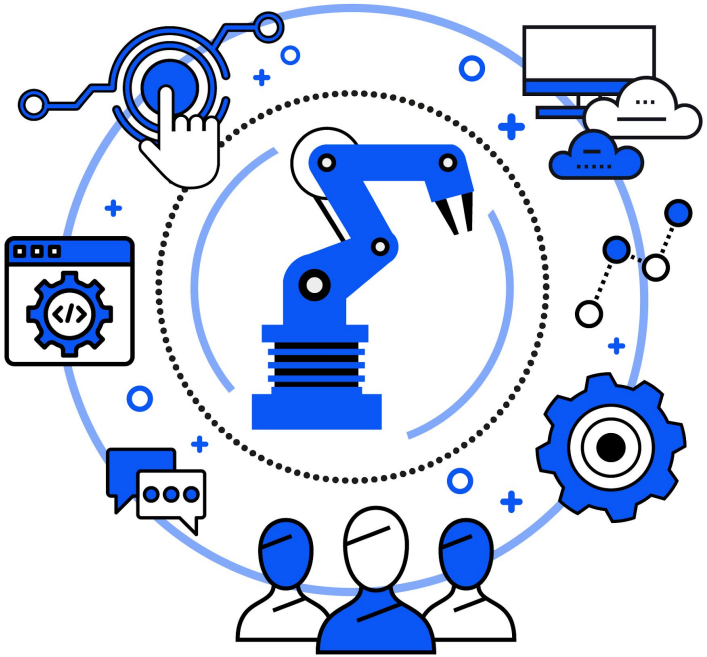


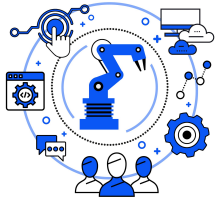
PICKNIK

Easy Robot Software

And the MoveIt! Setup Assistant 2.0

Dave Coleman, PhD
davetcoleman

Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study
David Coleman, Ioan Sucan, Sachin Chitta, Nikolaus Correll
Journal of Software Engineering for Robotics, April 2014



PICKNIK

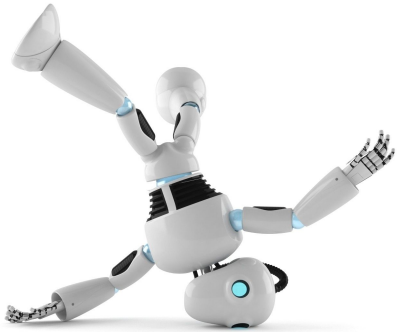
Outline

- Unique Challenges To Robot Software
- Why Do We Care?
- The 6 Entry Barrier Design Principles
- The MoveIt! Setup Assistant

PICKNIK

Building Robot Software Is Hard

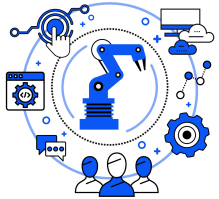
So let's all work together :)



PICKNIK

Unique Challenges Facing Robotic Software

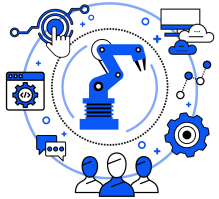
- No single developer can have the necessary **domain knowledge**
- Large variety in **complexity** and scale of robotic platforms.
- Software/hardware interaction with **unstructured real world**.
- Long term desire **reduce reliance on GUIs**.



PICKNIK

Barriers to Entry

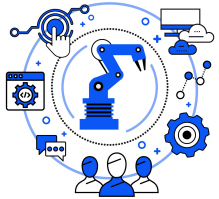
The time, effort, and knowledge that a new user must invest in the integration of a software component with an arbitrary robot.



Why do we care?

PICKNIK

Larger User Bases = Better Software

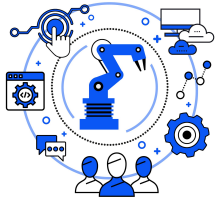


Why do we care? Larger user bases

PICKNIK

As number of users increases, bugs are identified and fixed faster [2]

[1] H. Bruyninckx, “Open robot control software: the orocos project,” in Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, vol. 3.
[2] D. C. Schmidt, “Why software reuse has failed and how to make it work for you,”C++ Report, vol. 11, no. 1, p. 1999
[3] D. C. Schmidt and A. Porter, “Leveraging open-source communities to improve the quality & performance of open-source software,” in Proceedings of the 1st Workshop on Open Source Software Engineering, 2001

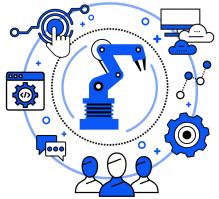


Why do we care? Larger user bases

PICKNIK

More users involved in quality assurance, documentation, and support [3]

[1] H. Bruyninckx, “Open robot control software: the orocos project,” in Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, vol. 3.
[2] D. C. Schmidt, “Why software reuse has failed and how to make it work for you,”C++ Report, vol. 11, no. 1, p. 1999
[3] D. C. Schmidt and A. Porter, “Leveraging open-source communities to improve the quality & performance of open-source software,” in Proceedings of the 1st Workshop on Open Source Software Engineering, 2001

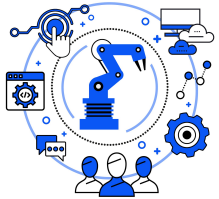


Why do we care? Larger user bases

PICKNIK

New feature contributions increase (weaker correlation) [2]

[1] H. Bruyninckx, “Open robot control software: the orocos project,” in Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, vol. 3.
[2] D. C. Schmidt, “Why software reuse has failed and how to make it work for you,”C++ Report, vol. 11, no. 1, p. 1999
[3] D. C. Schmidt and A. Porter, “Leveraging open-source communities to improve the quality & performance of open-source software,” in Proceedings of the 1st Workshop on Open Source Software Engineering, 2001

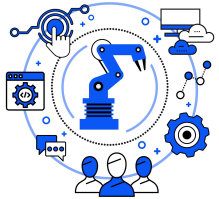


Why do we care? Larger user bases

PICKNIK

Critical mass of skilled contributors has been shown to make open source projects successful [1]

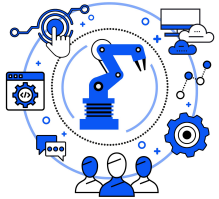
[1] H. Bruyninckx, “Open robot control software: the orocos project,” in Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, vol. 3.
[2] D. C. Schmidt, “Why software reuse has failed and how to make it work for you,”C++ Report, vol. 11, no. 1, p. 1999
[3] D. C. Schmidt and A. Porter, “Leveraging open-source communities to improve the quality & performance of open-source software,” in Proceedings of the 1st Workshop on Open Source Software Engineering, 2001



Why do we care? Hiring and Innovation.

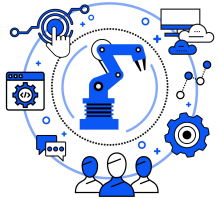
PICKNIK

Increase number of creative minds working on today's robotic challenge



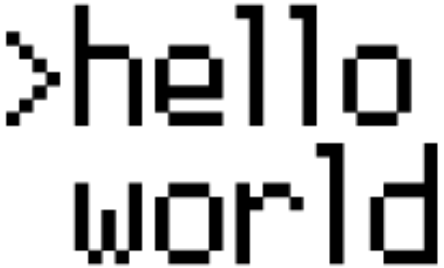
PICKNIK

The 6 Entry Barrier Design Principles



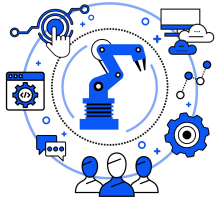
The 6 Entry Barrier Design Principles

Immediacy



Minimize the amount of time to accomplish the most basic task.

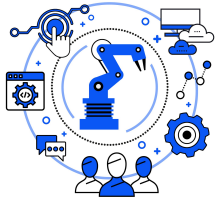
- Quick start demo
- Cursory feedback to new user that software is worth investing in
- Combat the paradox of the active user



Paradox of the Active User

Users never read manuals

The paradox is that the users would actually save time in the long run if they learned more about the system before attempting to use it, but these studies showed that in reality people do not tend to invest time upfront into learning a new system.

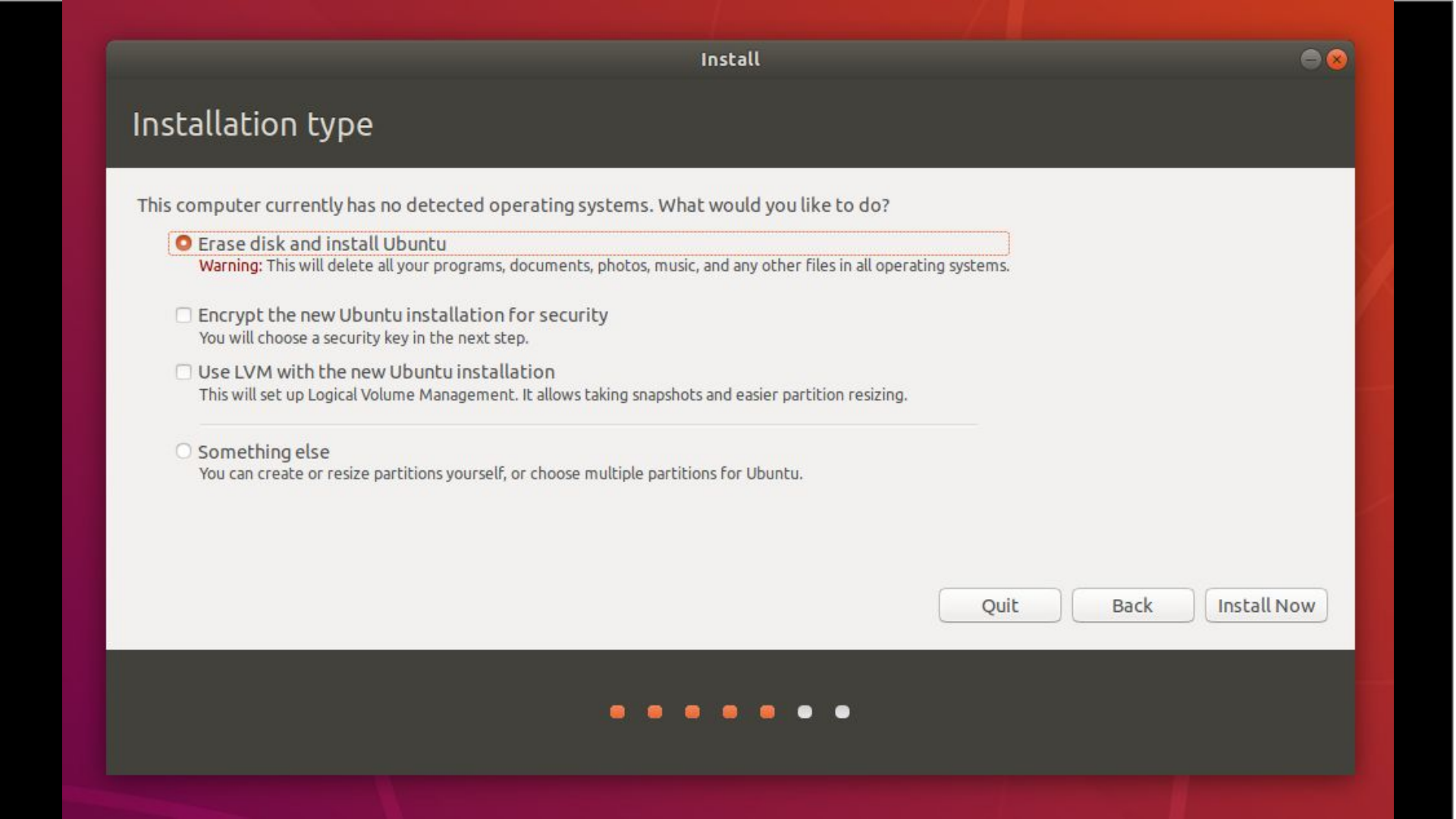


The 6 Entry Barrier Design Principles

Transparency

Configuration steps are performed automatically for the user while at the same time being as visible as possible

- Understand what parameters are specific to their robot
- "layered" approach of quick initial setup while allowing later customization as needed


Install
Installation type
This computer currently has no detected operating systems. What would you like to do?
Erase disk and install Ubuntu
Warning: This will delete all your programs, documents, photos, music, and any other files in all operating systems.
Encrypt the new Ubuntu installation for security
You will choose a security key in the next step.
Use LVM with the new Ubuntu installation
This will set up Logical Volume Management. It allows taking snapshots and easier partition resizing.
Something else
You can create or resize partitions yourself, or choose multiple partitions for Ubuntu.
Quit
Back
Install Now

The 6 Entry Barrier Design Principles

Reconfigurable

The automatically generated parameters and default values for the initial setup of a robot should be easy for the user to modify at a later time.

- Typically chosen to work with the largest number of robots
- Not optimal for any robot
- Varying applications require different configurations

The 6 Entry Barrier Design Principles

PICKNIK

Intuitive

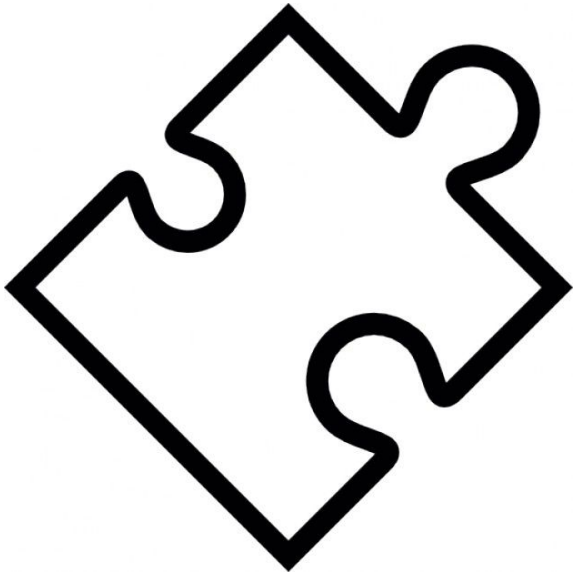
The need to read accompanied documentation, and the amount of required documentation, should be minimized.

- Follow standard design patterns
- Provide interface context clues
- Ideally an interface does not require additional documentation

The 6 Entry Barrier Design Principles

PICKNIK

Extensible



The user should be enabled to customize as many components and behaviors as possible within the reasonable scope of the software.

- Makes the software far more powerful and re-usable for varying use cases
- Plugin interface

The 6 Entry Barrier Design Principles

PICKNIK

Documented

The amount of reference material explaining how to use the software should be maximized for as many aspects and user levels as possible.

- No software is intuitive enough to not require documentation
- Different types of documentation are required for different types of users
  - Developers vs end-users

The 6 Entry Barrier Design Principles

PICKNIK

Documented

ROS.org

About | Support | Discussion Forum | Service Status | Q&A answers.ros.org

Search: Submit

Documentation | Browse Software | News | Download

dataspeed_pds

indigo | kinetic | melodic

Documentation Status

dataspeed_pds: *dataspeed_pds_can* | *dataspeed_pds_msgs* | *dataspeed_pds_scripts*

Package Summary

✔ Released ✔ Continuous Integration ✔ Documented

Interface to the Dataspeed Inc. Power Distribution System (PDS)

- Maintainer status: developed
- Maintainer: Kevin Hallenbeck <khallenbeck AT dataspeedinc DOT com>, Eric Myllyoja <emyllyoja AT dataspeedinc DOT com>
- Author: Kevin Hallenbeck <khallenbeck AT dataspeedinc DOT com>, Eric Myllyoja <emyllyoja AT dataspeedinc DOT com>
- License: BSD
- External website: http://dataspeedinc.com
- Bug / feature tracker: https://bitbucket.org/dataspeedinc/dataspeed_pds/issues
- Source: hg https://bitbucket.org/DataspeedInc/dataspeed_pds (branch: default)

Package Links
dataspeed_pds website
FAQ
Changelog
Change List
Reviews
Dependencies (4)
Jenkins jobs (11)

Wiki
Distributions
ROS/Installation
ROS/Tutorials
RecentChanges
dataspeed_pds

Page
Immutable Page
Info
Attachments
More Actions:

User
Login

Except where otherwise noted, the ROS wiki is licensed under the Creative Commons Attribution 3.0 | Find us on Google+

Wiki: dataspeed_pds (last edited 2018-07-02 18:43:10 by KevinHallenbeck)

Brought to you by: Open Source Robotics Foundation

dataspeed_pds

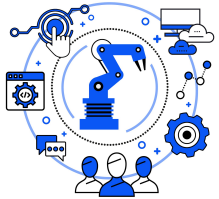
Overview
Source
Commits
Branches
Pull requests
Pipelines
Issues
Downloads

Dataspeed Inc. / Power Distribution / dataspeed_pds / Source

Source

default | dataspeed_pds /

dataspeed_pds		
dataspeed_pds_can		
dataspeed_pds_lcm		
dataspeed_pds_msgs		
dataspeed_pds_rqt		
dataspeed_pds_scripts		
.hgignore	377 B	2017-07-26
.hgtags	282 B	2018-06-29
LICENSE	1.6 KB	2018-04-12
bitbucket-pipelines.yml	2.4 KB	2018-06-28
dataspeed_pds.rosinstall	327 B	2017-07-26



MoveIt!

PICKNIK

MoveIt! Setup Assistant

Quick Setup of MoveIt!

PICKNIK

MoveIt! Setup Assistant

- Start
- Self-Collisions
- Virtual Joints
- Planning Groups
- Robot Poses
- End Effectors
- Passive Joints
- 3D Perception
- Simulation
- ROS Control
- Author Information
- Configuration Files

MoveIt! Setup Assistant

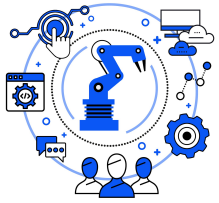
These tools will assist you in creating a Semantic Robot Description Format (SRDF) file, various yaml configuration and many roslaunch files for utilizing all aspects of MoveIt! functionality.

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



Optimize Collision Checking

PICKNIK

MoveIt! Setup Assistant

- Start
- Self-Collisions
- Virtual Joints
- Planning Groups
- Robot Poses
- End Effectors
- Passive Joints
- 3D Perception
- Simulation
- ROS Control
- Author Information
- Configuration Files

Optimize Self-Collision Checking

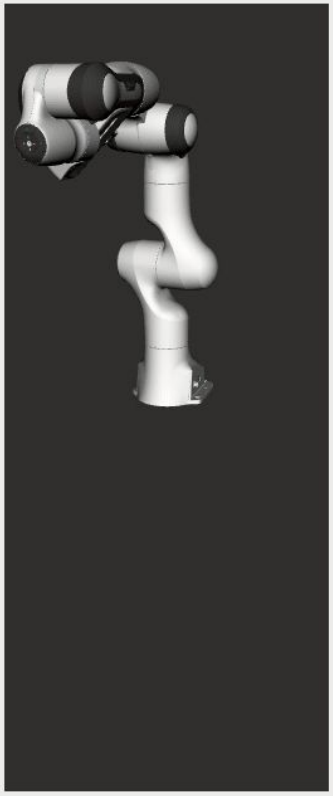
This searches for pairs of robot links that can safely be disabled from collision checking, decreasing motion planning time. These pairs are disabled when they are always in collision, never in collision, in collision in the robot's default position, or when the links are adjacent to each other on the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

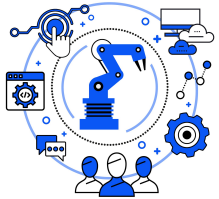
Sampling Density: Low — High 100000

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

	panda_link0	panda_link1	panda_link2	panda_link3	panda_link4	panda_link5	panda_link6	panda_link7	panda_hand	panda_leftfinger	panda_rightfinger
panda_link0		☑	☑	☑	☑	☐	☐	☐	☐	☐	☐
panda_link1	☑		☑	☑	☑	☐	☐	☐	☐	☐	☐
panda_link2	☑	☑		☑	☑	☐	☐	☐	☐	☐	☐
panda_link3	☑	☑	☑		☑	☑	☑	☑	☑	☑	☑
panda_link4	☑	☑	☑	☑		☑	☑	☑	☑	☑	☑
panda_link5	☐	☐	☐	☑	☑		☑	☑	☑	☐	☐
panda_link6	☐	☐	☐	☑	☑	☑		☑	☑	☑	☑
panda_link7	☐	☐	☐	☑	☑	☑	☑		☑	☑	☑
panda_hand	☐	☐	☐	☑	☑	☑	☑	☑		☑	☑

link name filter — ○ linear view ◉ matrix view — Revert

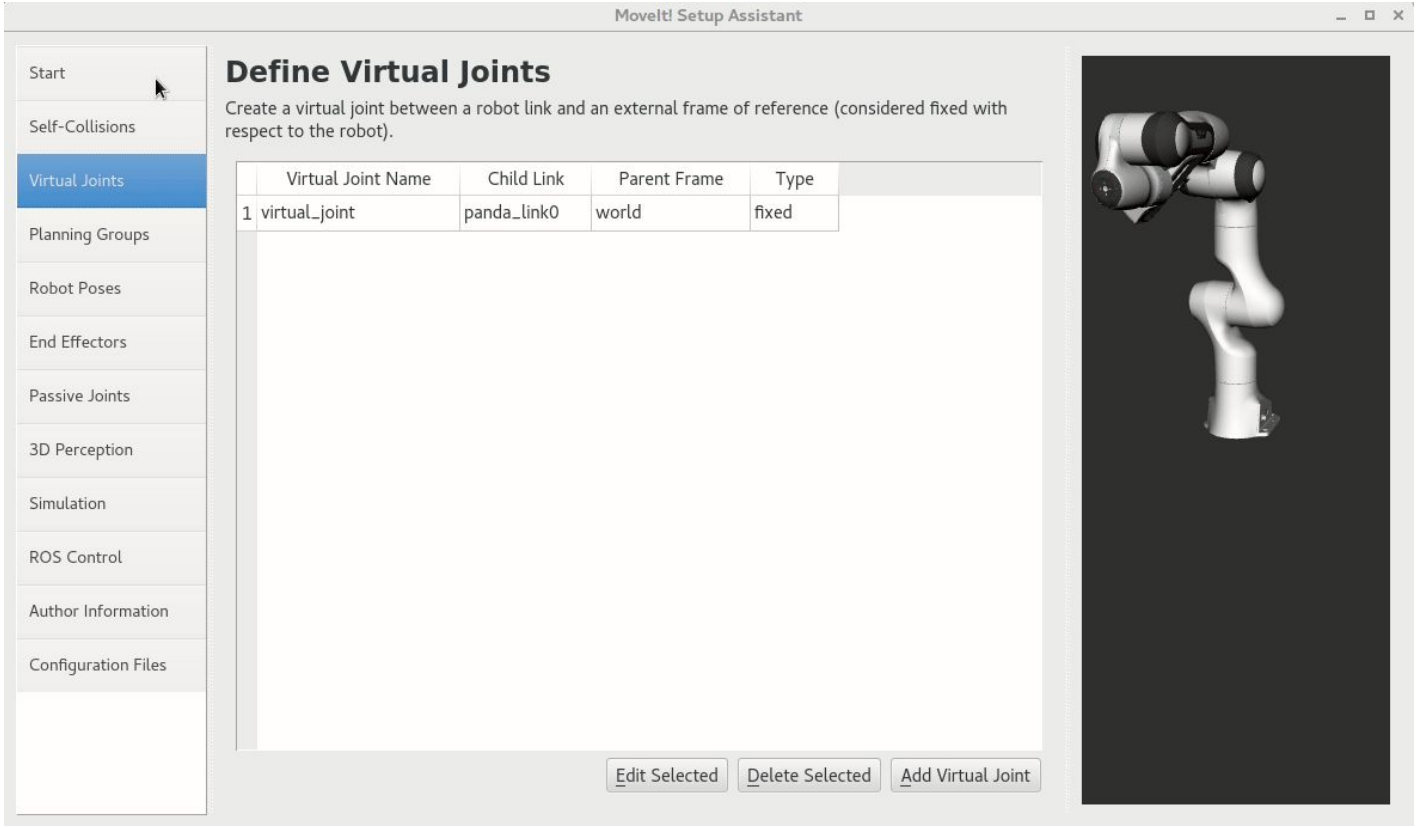


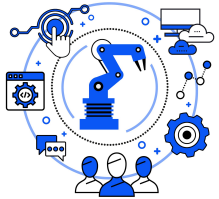


Specify Metadata

PICKNIK

- Planning Groups
- Robot Poses
- End Effectors
- Passive Joints
- Virtual Joints





MoveIt!

PICKNIK

MoveIt! Setup Assistant 2.0

Special thanks to Mohmmad El Khzragy, Open Robotics, and GSoC

Auto-configure Depth Sensors

PICKNIK

MoveIt! Setup Assistant

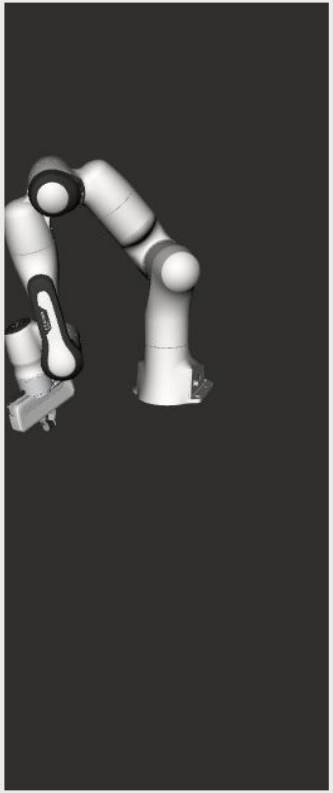
- Start
- Self-Collisions
- Virtual Joints
- Planning Groups
- Robot Poses
- End Effectors
- Passive Joints
- 3D Perception
- Simulation
- ROS Control
- Author Information
- Configuration Files

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



Setup Gazebo Simulation Integration

PICKNIK

Setup ROS Control

PICKNIK

MoveIt! Setup Assistant

- Start
- Self-Collisions
- Virtual Joints
- Planning Groups
- Robot Poses
- End Effectors
- Passive Joints
- 3D Perception
- Simulation
- ROS Control
- Author Information
- Configuration Files

Setup ROS Controllers

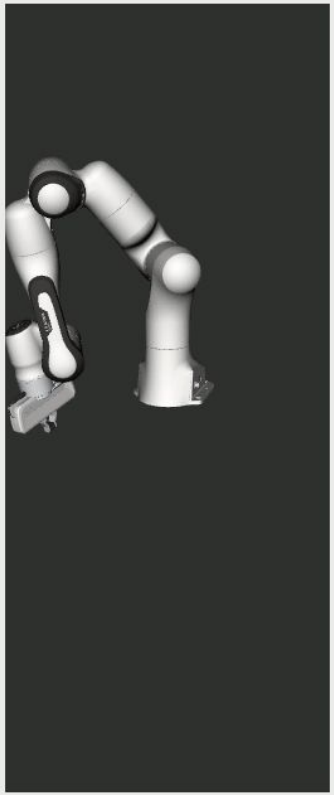
Configure MoveIt! to work with ROS Control to control the robot's physical hardware

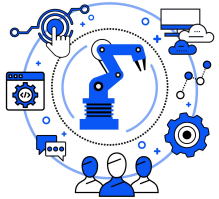
Auto Add FollowJointsTrajectory Controllers For Each Planning Group

Controller	Controller Type

Expand All Collapse All

Delete Controller | Add Controller | Edit Selected





MoveIt!

PICKNIK

Hello World

Generated Quick Start Demo

PICKNIK

moveit.rviz* - RViz

File Panels Help

Displays

Robot Alpha	0.5
Attached Body Color	150; 50; 150
Links	
Planning Request	
Planning Group	panda_arm_hand
Show Workspace	
Query Start State	
Query Goal State	
Interactive Marker Size	0
Start State Color	50; 230; 50
Start State Alpha	1
Goal State Color	246; 160; 49
Goal State Alpha	1
Colliding Link Color	255; 0; 0
Joint Violation Color	255; 0; 255

Add Duplicate Remove Rename

MotionPlanning

Context Planning Manipulation Scene Objects Stored Scenes Stored States Status

Commands: Plan, Execute, Plan and Execute, Stop

Query: Select Start State:, Select Goal State:, <random valid>, Update, Clear octomap

Options: Planning Time (s): 5.00, Planning Attempts: 10.00, Velocity Scaling: 1.00, Acceleration Scaling: 1.00, Allow Replanning, Allow Sensor Positioning, Allow External Comm., Use Collision-Aware IK, Allow Approx IK Solutions

Path Constraints: None, Goal Tolerance: 0.00

MotionPlanning - Trajectory Slider

Waypoint: 0 — 0 Pause

Reset **Left-Click:** Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Move Z. **Shift**: More options. 31 fps

New MoveIt! Tutorials

PICKNIK

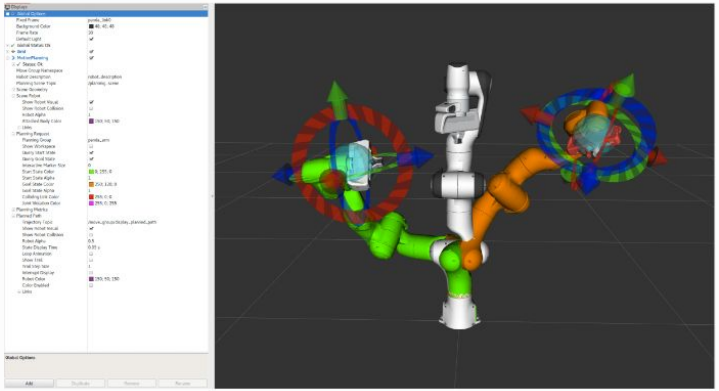
MoveIt!

MoveIt! Website
Blog

Docs » MoveIt! Quickstart in RViz

Edit on GitHub

MoveIt! Quickstart in RViz

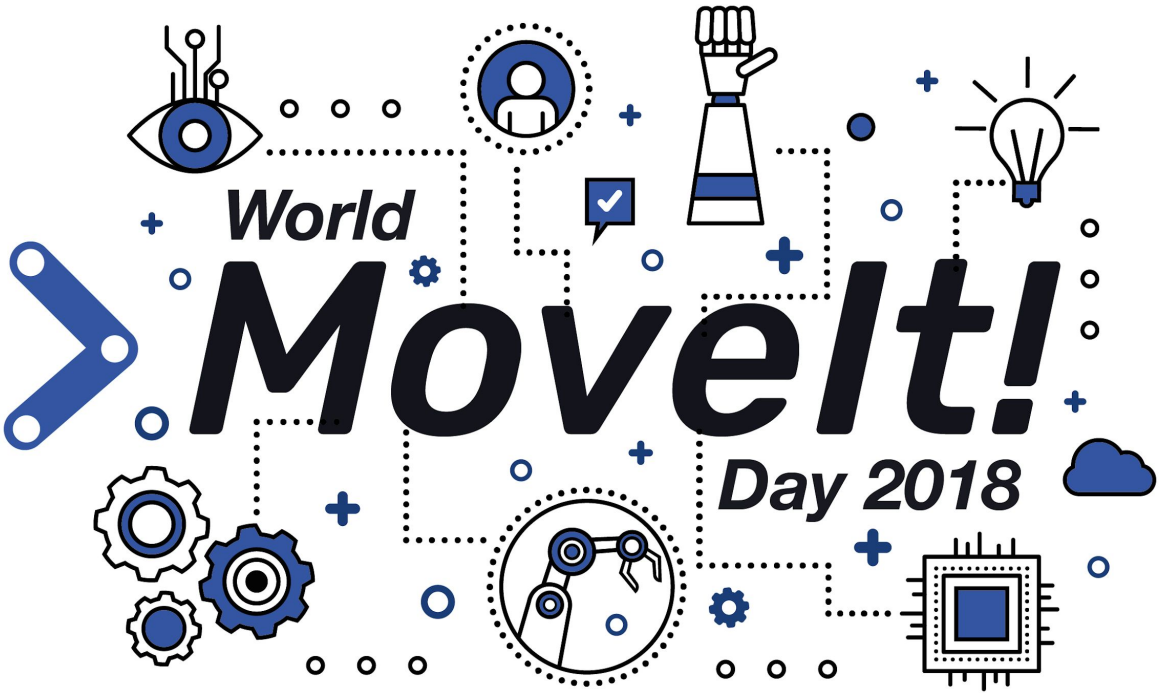


The quickest way to get started using MoveIt! is through its RViz plugin. Rviz is the primary visualizer in ROS and an incredibly useful tool for debugging robotics. The MoveIt! Rviz plugin allows you to setup virtual environments (scenes), create start and goal states for the robot interactively, test various motion planners, and visualize the output.

Getting Started

If you haven't already done so, make sure you've completed the steps in Getting Started.

Step 1: Launch the Demo and Configure the Plugin



Thursday, October 25th, 2018

Conclusion

PICKNIK

- Robot software is hard
- Make your software easier to use
- Apply the entry barrier design principles
- By building a community around your software, the software gets better
- The MoveIt! Setup Assistant is a good example