Movelt Overview

Mark Moll, PhD
Director of Research
PickNik Robotics

mamoll
Outline

- MoveIt Background
- MoveIt Applications: Land, Sea, Space
- Typical Usage Patterns
- Components
- More Resources
The Dream: A Multi Purpose Robot

Powered by MoveIt and ROS

Willow Garage
MoveIt: A Hardened Motion Planning Platform
## A Feature-Rich Ecosystem

<table>
<thead>
<tr>
<th>Global Planners</th>
<th>Cartesian Planners</th>
<th>Inverse Kinematics</th>
</tr>
</thead>
<tbody>
<tr>
<td>● OMPL</td>
<td>● RobotState</td>
<td>● KDL</td>
</tr>
<tr>
<td>● SBPL</td>
<td>● Descartes</td>
<td>● IKFast</td>
</tr>
<tr>
<td>● TrajOpt</td>
<td>● JogArm</td>
<td>● TracIK</td>
</tr>
<tr>
<td>● STOMP</td>
<td>● PilzMotion</td>
<td>● LMA</td>
</tr>
<tr>
<td>● CHOMP</td>
<td></td>
<td>● BioIK</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Grasping Libraries</th>
<th>Collision Checking</th>
<th>Perception / Octomap</th>
</tr>
</thead>
<tbody>
<tr>
<td>● MoveIt Grasps</td>
<td>● Fast Collision</td>
<td>● Depth Images</td>
</tr>
<tr>
<td>● Grasp Pose</td>
<td>Library (FCL)</td>
<td>● Point Clouds</td>
</tr>
<tr>
<td>Detection (GPD)</td>
<td>● Bullet</td>
<td></td>
</tr>
<tr>
<td>● Intel OpenVino</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPD</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Flexibility (but also complexity)

- Can handle:
  - Groups of joints
  - Multivariable joints
  - Mimic joints

- Notions of:
  - Cartesian-Space Planning
  - Joint-Space Planning
  - Orientation Constraints
  - Visibility Constraints
Differential drive treated as a special joint:

- 3 DOFs: position + orientation
- joint distance/interpolation overridden to correspond to optimal diff-drive paths
- rest of MoveIt “just works”
152 Robot types integrated to work with MoveIt

29,843 Downloads per month of moveit_core

733 Academic citations of MoveIt

162,630 Unique users to moveit.ros.org in 2021

5,600 Members of Discourse, MoveIt's Discussion Forum

1,136 Github users have starred the MoveIt project

262 Github code contributors to MoveIt

167 International attendees of 2020 MoveItWorld online event
An Active Community

- World MoveIt Day (annually)
- Monthly MoveIt Manipulation Working Group meetings
- ROSCON / ROS World workshops & tutorials
Land, Sea, Space
Sea

Remotely Operated Underwater Vehicles

ROVs
Space

IVR
Inter Vehicle Robotics

OSAM
On-orbit Servicing, Assembly, and Manufacturing

Planetary Activities
Lunar Base Construction
Typical Usage Patterns
Setup Assistant  *(work in progress for MoveIt 2)*

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

![Collision Matrix Example](image)
Rviz Motion Planning Plugin
ROS Services & Actions

User Interface
- move_group_interface (C++)
- moveit_commander (Python)
- GUI (Rviz Plugin)
- Other Interfaces

move_group
- MoveGroupAction
- PickAction
- PlaceAction
- Get CartesianPath Service
- Get IK Service
- Get FK Service
- Get Plan Validity Service
- Plan Path Service
- Execute Path Service
- Get Planning Scene Service
- AttachedObject
- CollisionObject
- PlanningSceneDiff

ROS Param Server
- UNDF
- SRDF
- Config

Robot Controllers
- Joint Trajectory
- Point Cloud

Robot Sensors
- Robot State Publisher
- TF

Robot 3D Sensors
Pro-Tip: Use C++ classes individually

```
robot_model_loader_.reset(new robot_model_loader::RobotModelLoader("robot_description"));
robot_model_ = robot_model_loader_->getModel();
planning_scene_.reset(new planning_scene::PlanningScene(robot_model_));
tf_.reset(new tf::TransformListener(nh_));
psm_.reset(new planning_scene_monitor::PlanningSceneMonitor(
    planning_scene_, robot_model_loader_, tf_, "my_scene");
psm_->startStateMonitor("/joint_states", "");
psm_->startPublishingPlanningScene(planning_scene_monitor::PlanningSceneMonitor::
    UPDATE_SCENE, "my_planning_scene");
visuals_tools_.reset(new MoveItVisualTools(robot_model_, planning_scene_monitor_));
planning_pipeline_.reset(new planning_pipeline::PlanningPipeline(robot_model_nh_,
    "planning_plugin", "request_adapters");
trajectory_execution_manager_.reset(new trajectory_execution_manager::
    TrajectoryExecutionManager(robot_model_));
```

**Best for researchers who want to modify code**
```cpp
static const std::string PLANNING_GROUP = "stretch_arm";
auto moveit_cpp_ptr = std::make_shared<moveit_cpp::MoveItCpp>(node);
moveit_cpp_ptr->getPlanningSceneMonitor()->providePlanningSceneService();
auto planning_components = std::make_shared<moveit_cpp::PlanningComponent>(PLANNING_GROUP, moveit_cpp_ptr);
planning_components->setStartStateToCurrentState();
planning_components->setGoal("extended");
auto plan_solution = planning_components->plan();

if (plan_solution)
{
  |
  |
}
```

*Best for researchers who want to modify code*
Movelt Task Constructor
Components
High Level Components

- High Level Tasks / AI
- Graphical Interfaces
- Benchmarking
- Configuration
- Motion Planning Framework
- Sensor Data
- Planning Scene
- Joint States
- Trajectory Execution
- Planning Algorithms
- Controllers
- Forward & Inverse Kinematics
- Collision Detection
Plugin Framework

High Level Capability Plugins

Kinematics Plugins

Collision Detection Plugins

Controller Manager Plugins

Robot Model

Joint States

Robot Perception Sensors

Octomap Updater Plugins

Planning Plugins

Planning Request Adapter Plugins

Robot Actuators

Full details here
Planning Scene Monitor

User Interface

- move_group_interface (C++)
- moveit_commander (Python)
- GUI (Rviz Plugin)
- Other Interfaces

Planning Scene Monitor

- AttachedObject
- CollisionObject
- PlanningSceneDiff

- Monitored Planning Scene (Optionally Published)

Robot Sensors

- 3D Sensors
- Robot Sensors
- Joint States Topic
- TF
- Robot State Publisher
- Point Cloud Topic
- Depth Image Topic
# Motion Planner Plugins

<table>
<thead>
<tr>
<th>OMPL</th>
<th>STOMP / CHOMP</th>
<th>TrajOpt</th>
<th>SBPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kavraki Lab at Rice</td>
<td>Kalakrishnan et al</td>
<td>John Schulman... Pieter Abbeel</td>
<td>Maxim Likhachev's Lab at CMU</td>
</tr>
<tr>
<td>Probabilistic, Sampling-Based</td>
<td>Optimization-Based</td>
<td>Sequential Convex Optimization</td>
<td>Graph-Based</td>
</tr>
<tr>
<td>Probabilistically Complete</td>
<td>Locally Optimal</td>
<td>Locally Optimal</td>
<td>Resolution Complete</td>
</tr>
<tr>
<td>Stochastic</td>
<td>Deterministic</td>
<td>Deterministic</td>
<td>Deterministic</td>
</tr>
</tbody>
</table>

- **Is a library of many planners**
- **Computationally fast**
- **More reliable runtime for real-world applications**
- **Many variants of algorithms available**
- **Generates smooth well behaved collision free motion paths in reasonable time**
- **Can incorporate additional objective functions - collision avoidance and smoothness**
- **Currently in final development**
- **Requires pre-processing phase**
- **Requires custom heuristics for different configuration spaces**
- **More reliable solutions for real-world applications**
Planner Request Adapters

- Plugins for pre-/post-processing planner request

- Two types of plugins:
  - Trajectory generation: (re)parametrize trajectory to respect velocity, acceleration, jerk limits.
  - Heuristic fixes for common problems:
    - set workspace bounds if none defined
    - tweak joints in start state to be within joint limits
    - tweak start state to be collision free

Adapts research theory to real world hardware
Inverse Kinematics

- **KDL** - x dof
  - Kinematics Dynamics Library
  - OROCOS
- **IKFast** - 6 or 7 dof
  - OpenRave Analytical
- **Trac-IK**
  - Combines KDL with Sequential Quadratic Programming approach
- **OPW Kinematics** - 6 dof
  - Closed form IK for parallel base, spherical wrist industrial manipulators
Time Parameterization

- TOTG: Time Optimal Trajectory Generation (recommended)
- Ruckig: Only method that supports jerk limits (new!)
- ITPP: Iterative Time Path Parameterization
- TOPP: Time Optimal Path Parameterization

More info [here](#)
## Comparison of MoveIt 1 and MoveIt 2

<table>
<thead>
<tr>
<th>Feature</th>
<th>MoveIt 1</th>
<th>MoveIt 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROS 1 Support</td>
<td>✓</td>
<td>✓ via rost_bridge</td>
</tr>
<tr>
<td>ROS 2 Support</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Motion Planning</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Inverse Kinematics</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Perception</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Grasping</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Setup Assistant</td>
<td>✓</td>
<td>✓ in development</td>
</tr>
<tr>
<td>MoveIt Task Constructor</td>
<td>✓</td>
<td>✓ pending</td>
</tr>
<tr>
<td>Game Controller Integration for Servo</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Industrial Trajectory Generator</td>
<td></td>
<td>✓ planned</td>
</tr>
<tr>
<td>Probabilistically complete Cartesian Planning</td>
<td></td>
<td>✓ State Patch</td>
</tr>
<tr>
<td>Composable Nodes</td>
<td></td>
<td>✓ nodelet subsystem</td>
</tr>
<tr>
<td>Planning for Differential Drive Bases</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Hybrid Planning (global + local planners)</td>
<td></td>
<td>✓ pending</td>
</tr>
<tr>
<td>Based on Realtime Capable DDS Messaging</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Native Windows Build</td>
<td></td>
<td>✓ via RoboStack</td>
</tr>
<tr>
<td>New Feature Development by PickNik</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Development Coordinated with ROS 2 Technical Steering Committee</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Built for Industrial Security</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
More Resources
Online Tutorials (http://moveit2_tutorials.picknik.ai)

MoveIt 2 Tutorials

These tutorials will quickly get you, and your robot, using the MoveIt 2 Motion Planning Framework.

In these tutorials, the Franka Emika Panda robot is used as a quick-start demo. Alternatively, you can easily use any robot that has already been configured to work with MoveIt - check the list of robots running MoveIt to see whether MoveIt is already available for your robot. Otherwise, you can setup MoveIt to work with your custom robot in the tutorial section "Integration with a New Robot", below.
Main website (http://moveit.ros.org)

Concepts
The following is an overview of how MoveIt works. For more concepts and details see the tutorials or the developers' concepts.

System Architecture
Quick High Level Diagram
Thanks!

PickNik Robotics
picknik.ai
Colorado, USA
@picknikrobotics

Mark Moll
mark@picknik.ai