The Descartes Planning Library for Semi-Constrained Cartesian Trajectories

Principle Engineer

Shaun Edwards

Southwest Research Institute

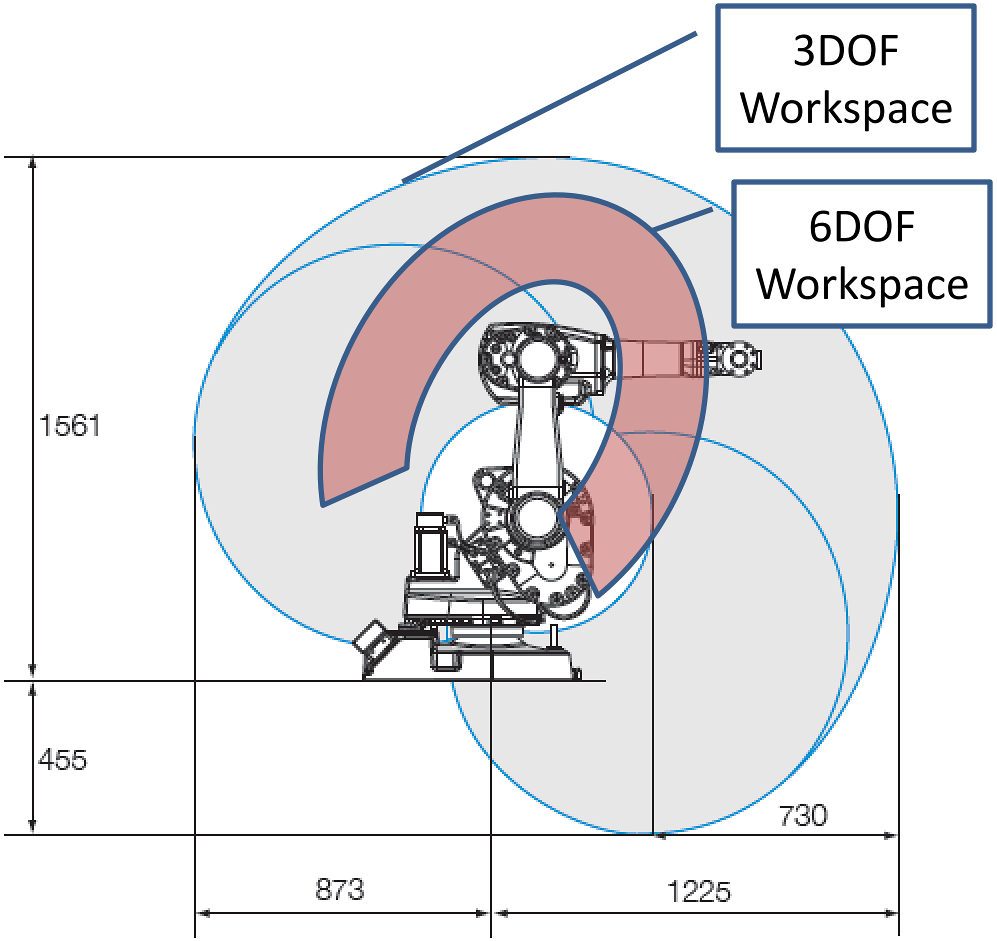
Application Need

- Semi-constrained trajectories: traj. DOF < robot DOF

Current Solution

- Arbitrary assignment of 6DOF poses, redundant axes -> IK
- Limited guarantee on trajectory timing
- Limitations
  - Reduced workspace
  - Relies on human intuition
  - Collisions, singularities,

  joint limits

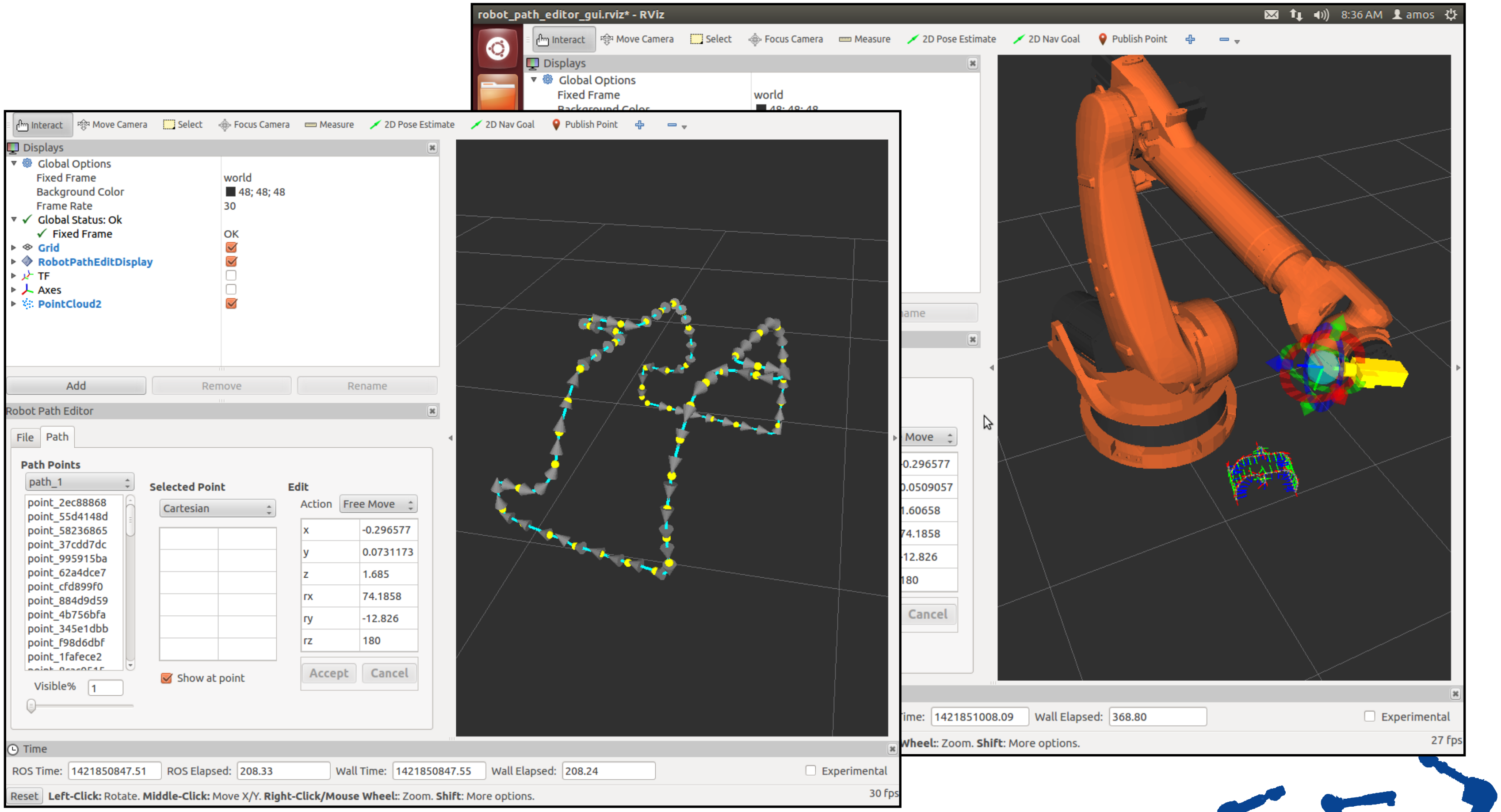


Descartes

- Planning library for semi-constrained trajectories
- Requirements
  - Generate common sense plans
  - Find easy solutions fast, hard solutions with time
  - Handle hybrid trajectories (joint, Cartesian, specialized points)
  - Fast re-planning/cached planning

Descartes Example

- Robotic Routing



robot_path_editor_gui.rviz* - RViz 8:36 AM amos

Displays

Global Options	
Fixed Frame	world
Background Color	■ 48; 48; 48
Frame Rate	30
✓ Global Status: Ok	
✓ Fixed Frame	OK
Grid	☑
RobotPathEditDisplay	☑
TF	□
Axes	□
PointCloud2	☑

Robot Path Editor

Path Points

path_1

point_2ec88868
point_55d4148d
point_58236865
point_37cdd7dc
point_995915ba
point_62a4dce7
point_cfd899f0
point_884d9d59
point_4b756bfa
point_345e1dbb
point_f98d6dbf
point_1fafece2

Visible% 1

Selected Point

Cartesian

☑ Show at point

Edit

Action Free Move

x	-0.296577
y	0.0731173
z	1.685
rx	74.1858
ry	-12.826
rz	180

Time

ROS Time: 1421850847.51 ROS Elapsed: 208.33 Wall Time: 1421850847.55 Wall Elapsed: 208.24 □ Experimental

Left-Click: Rotate. **Middle-Click**: Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options. 30 fps

-0.296577
0.0509057
1.60658
74.1858
-12.826
180

ime: 1421851008.09 Wall Elapsed: 368.80 □ Experimental

Wheel:: Zoom. **Shift**: More options. 27 fps

Performance

- 800 5DOF (6DOF robot) waypoints – ~~30s~~ 1-10s plan time
- Path planners minimize joint motion
- Re-planning near instantaneous
- Support for hybrid, kinematic planning

Robotic Routing

https://www.youtube.com/watch?v=cZxt00uoyBo

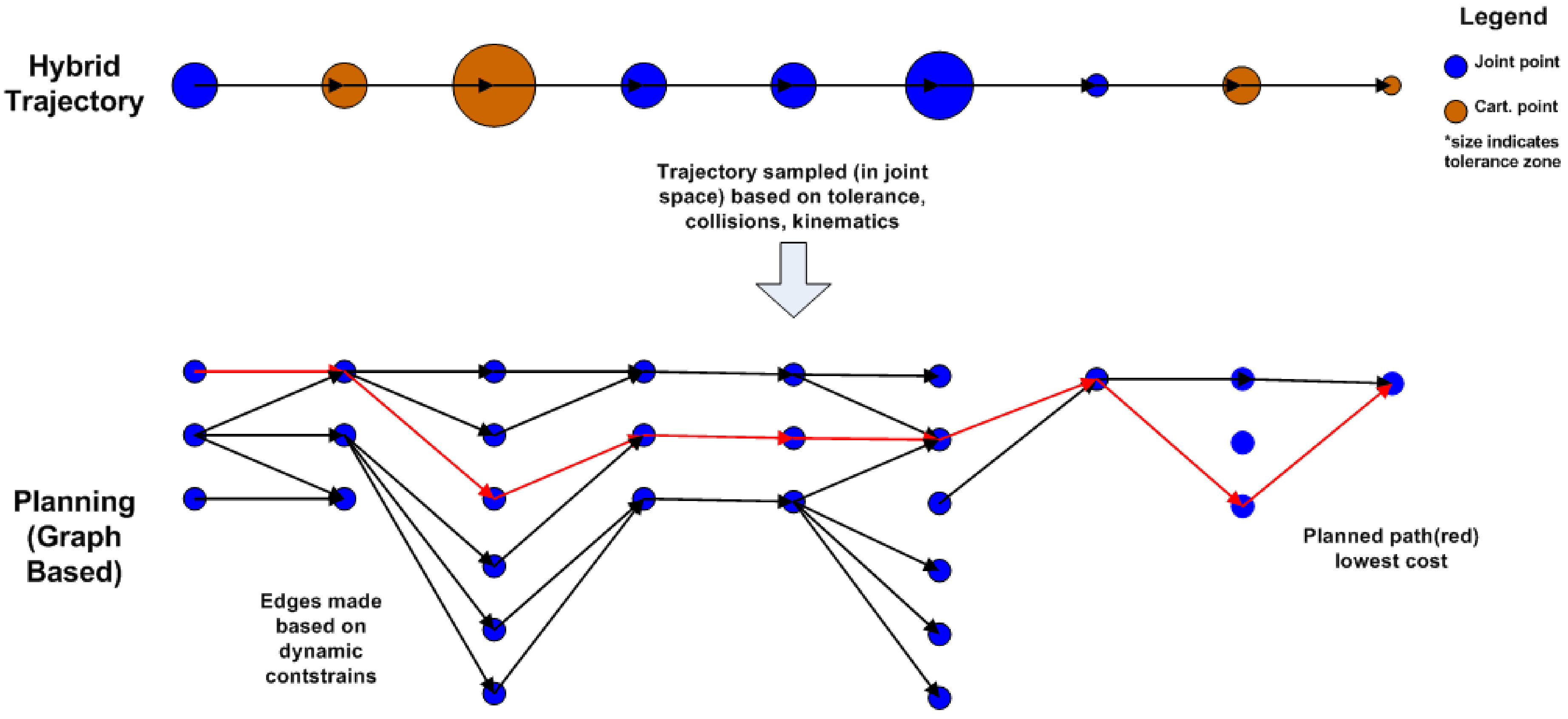
Descartes Interfaces

- Trajectory Point
  - Robot independent
  - Tolerance (fuzzy)
  - Timing
- Robot Model
  - IK/FK
  - Validity (Collision checking, limits)
  - Similar to MoveIt::RobotState, but with **getAllIK**
- Planner
  - Trajectory solving
  - Plan caching/re-planning

Descartes Implementations

- Trajectory Points
  - Cartesian point
  - Joint point
  - AxialSymmetric point (5DOF)
- Robot Model
  - MoveIt wrapper (working with MoveIt to make better)
  - FastIK wrappers
  - Custom solution
- Planners
  - Dense – graph based search
  - Sparse – hybrid graph based/interpolated search

Descartes Implementations



Hybrid Trajectory

Legend

● Joint point

● Cart. point

*size indicates tolerance zone

Trajectory sampled (in joint space) based on tolerance, collisions, kinematics

Planning (Graph Based)

Edges made based on dynamic contstrains

Planned path(red) lowest cost

Open Source Details

- Public development: https://github.com/ros-industrial-consortium/descartes
- Documentation: http://wiki.ros.org/descartes
- Releases
  - Hydro (stable) – binary
  - Indigo (unstable) – source
- Tutorials: http://wiki.ros.org/descartes/Tutorials
- Acknowledgements:
  - Supported by: NIST (70NANB14H226), ROS-Industrial Consortium FTP

Contact Information

Shaun Edwards

Southwest Research Institute

Contact

Phone: 210-522-3277
Email: sedwards@swri.org