Bench-MR: A Motion Planning Benchmark for Wheeled Mobile Robots

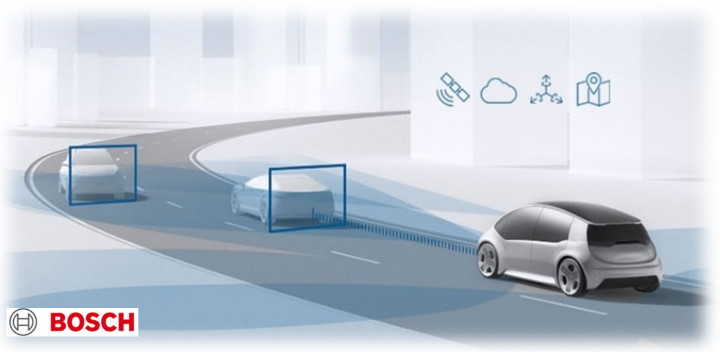
https://robot-motion.github.io/bench-mr

Eric Heiden[1], Luigi Palmieri[2], Leonard Bruns[3],
Kai O. Arras[2], Gaurav S. Sukhatme[1], Sven Koenig[1]

[1] Dept. of Computer Science, University of Southern California, USA
[2] Robert Bosch GmbH, Corporate Research, Germany
[3] Division of Robotics, Perception and Learning (RPL), KTH, Sweden

IEEE 2021 ICRA
May 30 to June 5, 2021
Xi'an China

Motivation



Benchmarking is crucial to evaluate progress in motion planning research, find a suitable combination of motion planning components for a particular application

Lack of specialized benchmarks for nonholonomic, wheeled mobile robots

Bench-MR:

- Applications in service and intralogistics robotics, autonomous driving, etc.
- Easy to use (high-level Python interface)
- Expandable (support for different back-ends, e.g. OMPL, SBPL)
- Feature-rich (various environment types, procedurally generated scenarios, planners, extend functions, etc.)

Main Related Work

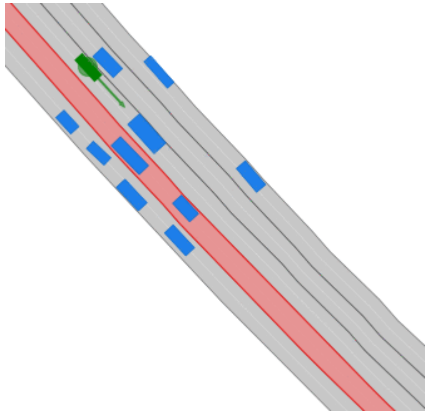
OMPL benchmark (Moll et al., RAL 2015)

- Benchmarking framework for generic robotic systems
- Not providing environments designed for wheeled mobile robots' applications

MovingAI (Sturtevant, TCIAIG 2012)

- Focusing on path findings on grids
- Not considering robot kinodynamic constraints

CommonRoad (Althoff et al., IVS 2017)



- Provides several automated driving scenarios
- Only lanes environments

Bench-MR

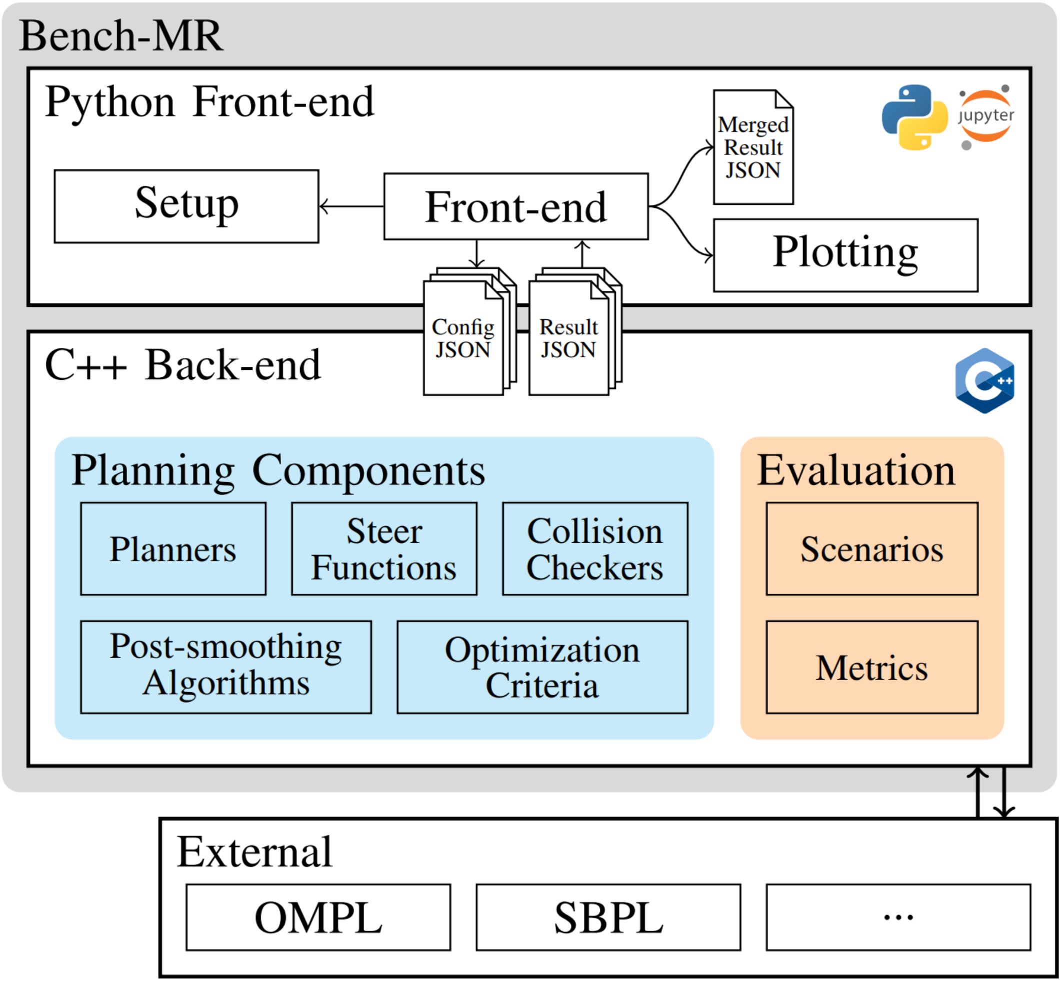
Motion Planning Benchmark for Wheeled Mobile Robots

Architecture

C++ back-end implements interfaces to planning libraries, environment types, etc.

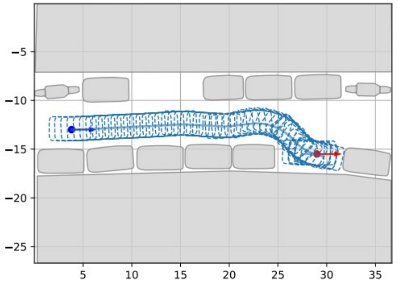
Front-end based on Jupyter notebooks provides high-level interface to set up benchmarks, evaluate results

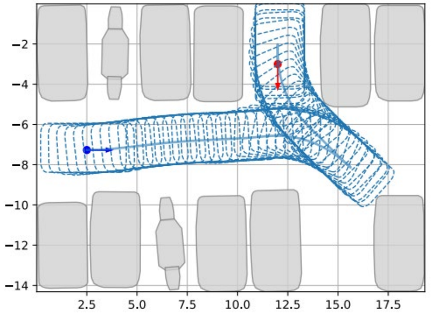
Configuration through JSON files ensures repeatability of experiments



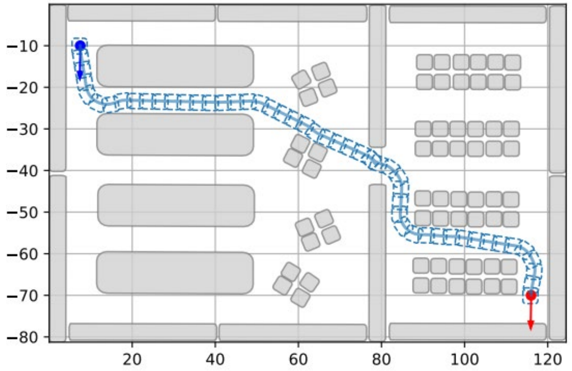
Environments

Polygon-based

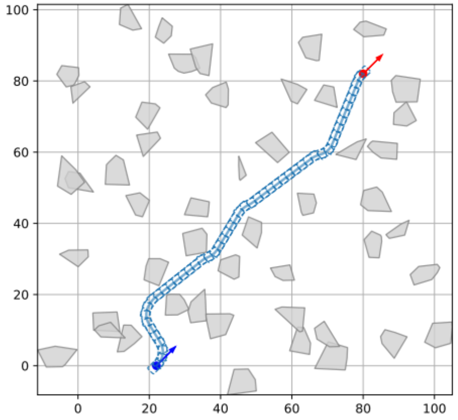




Parking scenarios



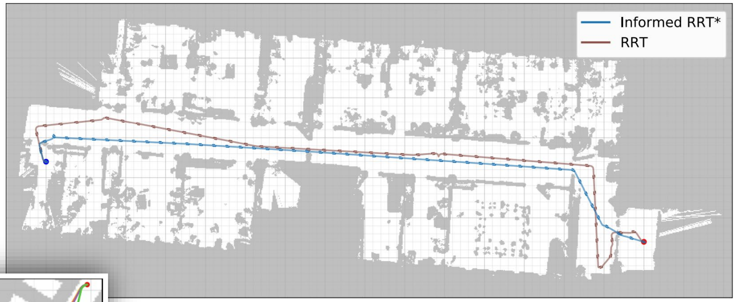
Warehouse navigation

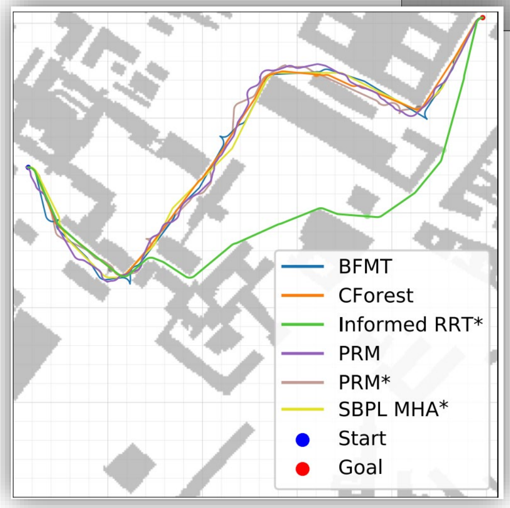


Procedurally generated

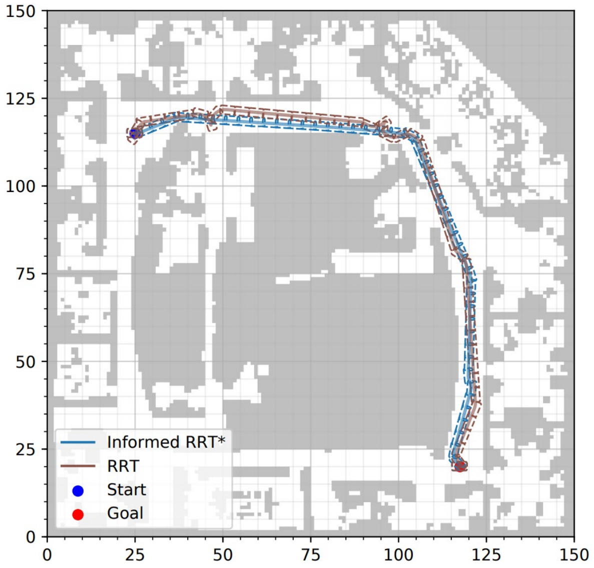
Grid-based

Occupancy grids from images



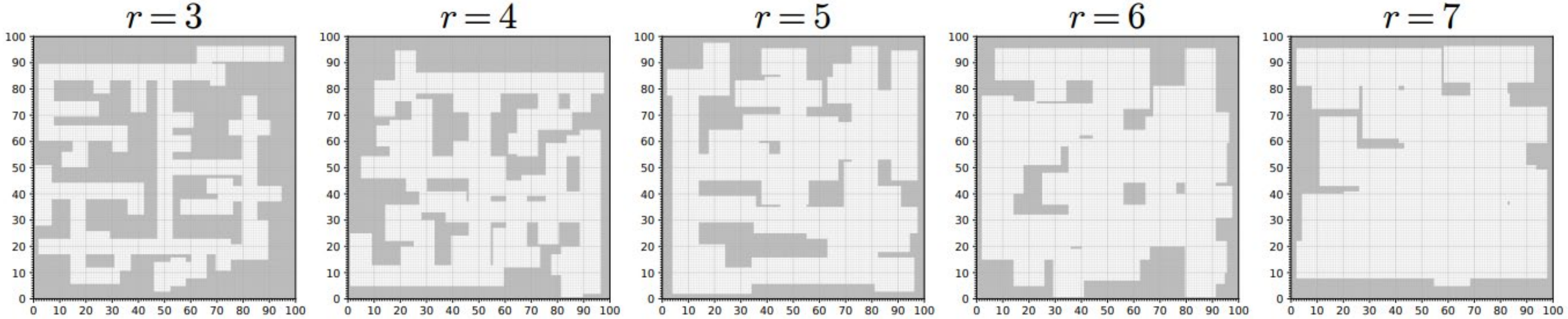


Moving AI Cities dataset

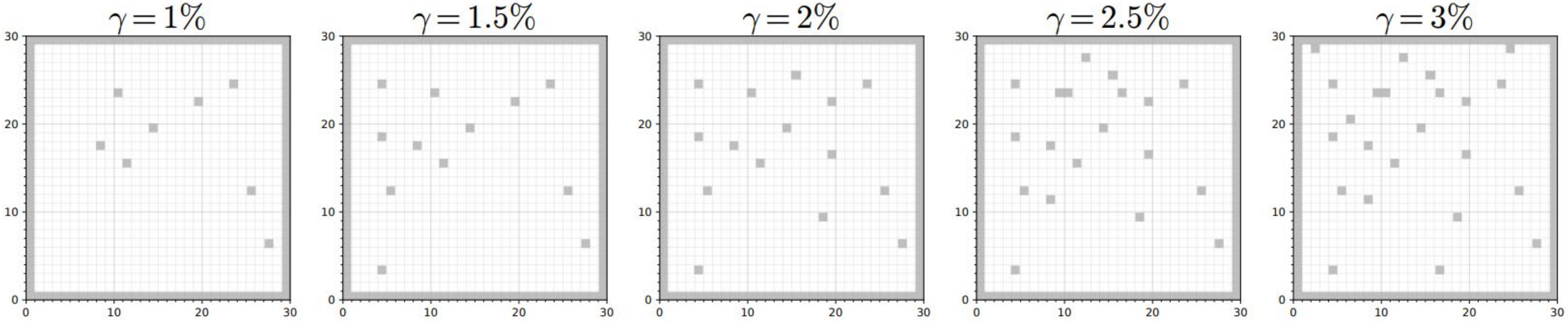


Procedurally Generated Environments

Corridor-like environments with defined corridor width

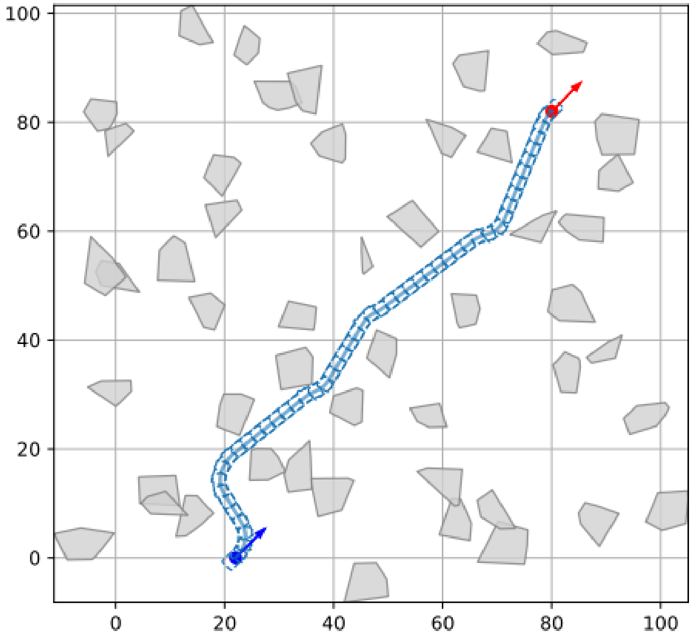


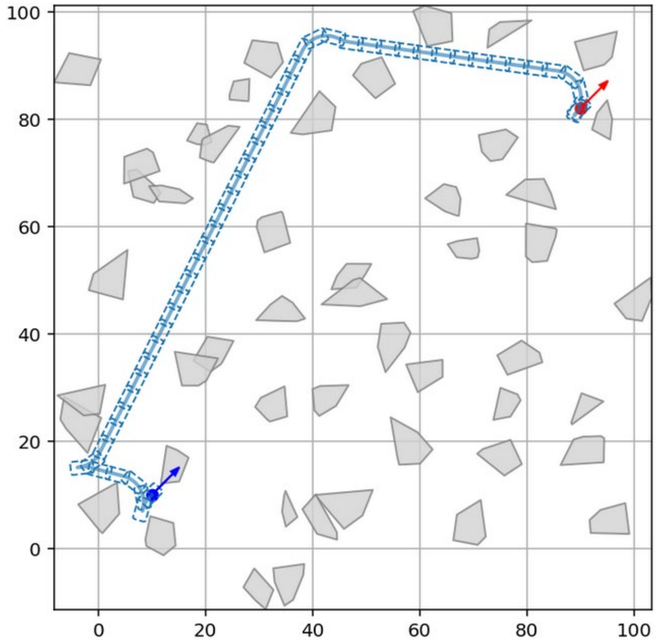
Random grids with defined occupancy ratio

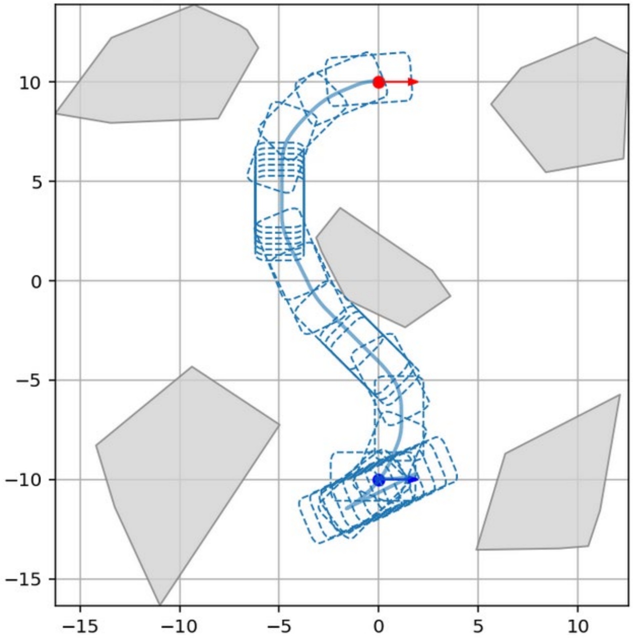


Procedurally Generated Environments

"Asteroid field" consisting of randomly generated convex polygons







Motion Planners

Sampling-based planners

- **Feasible planners**

  RRT, PRM, SPARS, EST, SBL, STRIDE, …

- **Asymptotically (near) optimal planners**

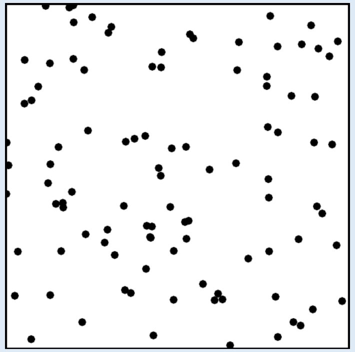
  RRT*, PRM*, BFMT, RRT$^{\#}$, Informed RRT*, CForest, …

Support for sampling from uniform distribution and deterministic sequences (e.g. Halton)

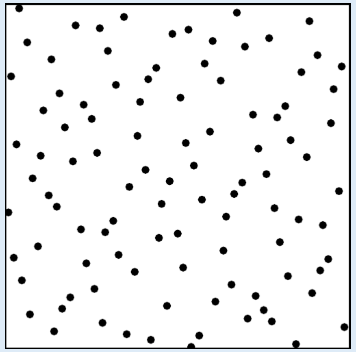
Open Motion Planning Library OMPL

(Sucan, Moll, Kavraki 2012)

Uniform



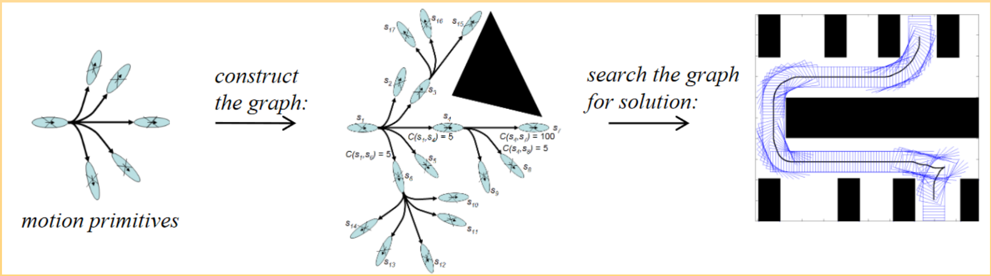
Halton



Lattice-based planners

ARA*, AD*,

MHA*, ANA*



SBPL

Search-Based Planning Library

(Likhachev et al. 2003)

Extend Functions

Extend functions connect consecutive states on a path

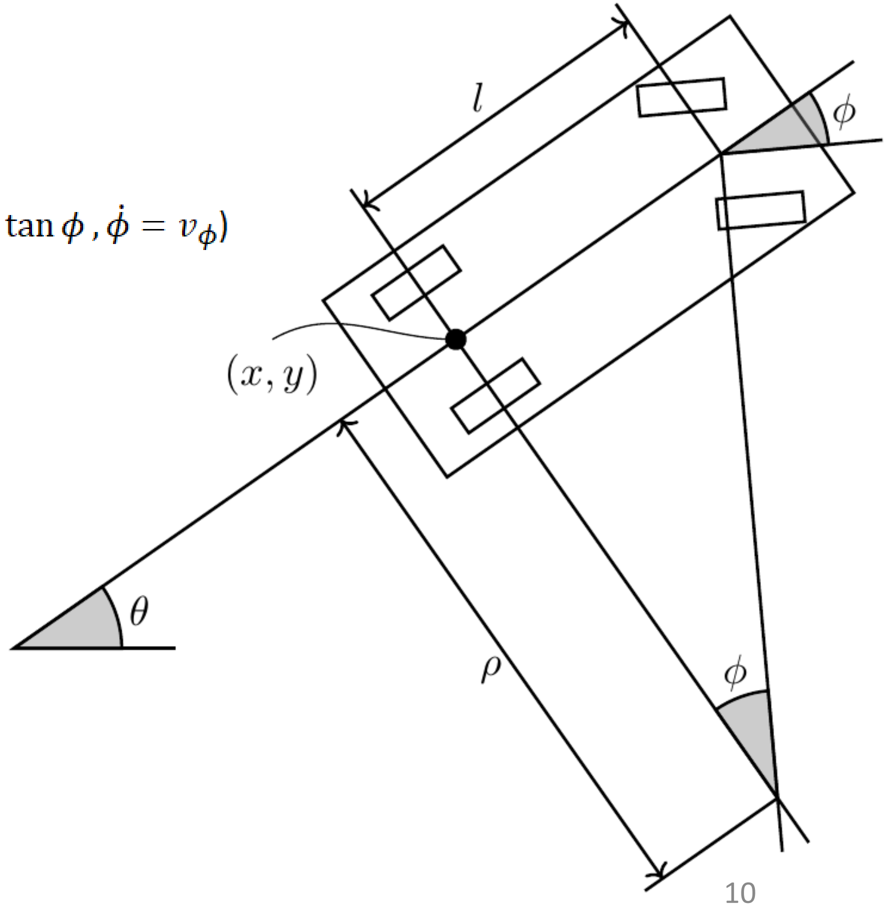
Robot models:

- Kinematic car $(\dot{x} = v\cos\theta,\ \dot{y} = v\sin\theta,\ \dot{\theta} = {}^{v}/_{l}\tan\phi)$
- Kinematic single-track model $(\dot{x} = v\cos\theta,\ \dot{y} = v\sin\theta,\ \dot{\theta} = {}^{v}/_{l}\tan\phi\,, \dot{\phi} = v_{\phi})$

Steer functions:

- Dubins, Reeds-Shepp
- Continuous Curvature
- POSQ

Motion primitives (SBPL)

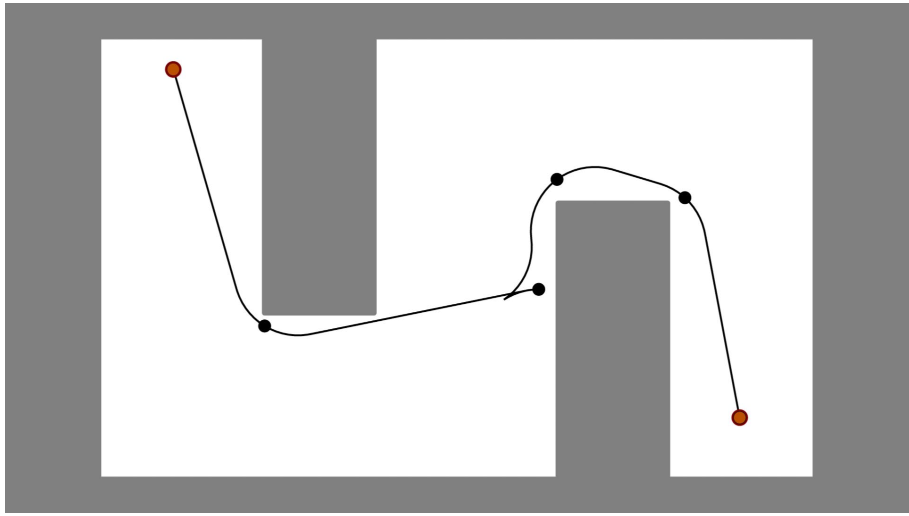


Post Smoothing Methods

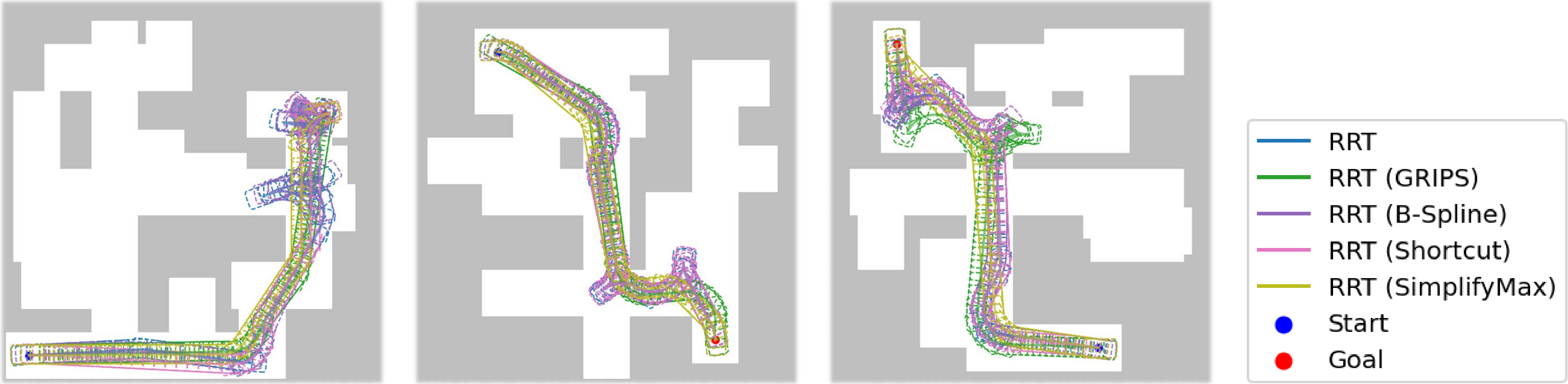
Smoothen path found by a motion planner

B-Spline*	fit B-spline through vertices to smoothen the path
Shortcut*	skip vertices on path to connect directly
Simplify Max*	combine B-spline + shortcut
GRIPS	move vertices on distance field + shortcut

* from OMPL



Gradient-informed path smoothing (GRIPS)



Optimization Objectives

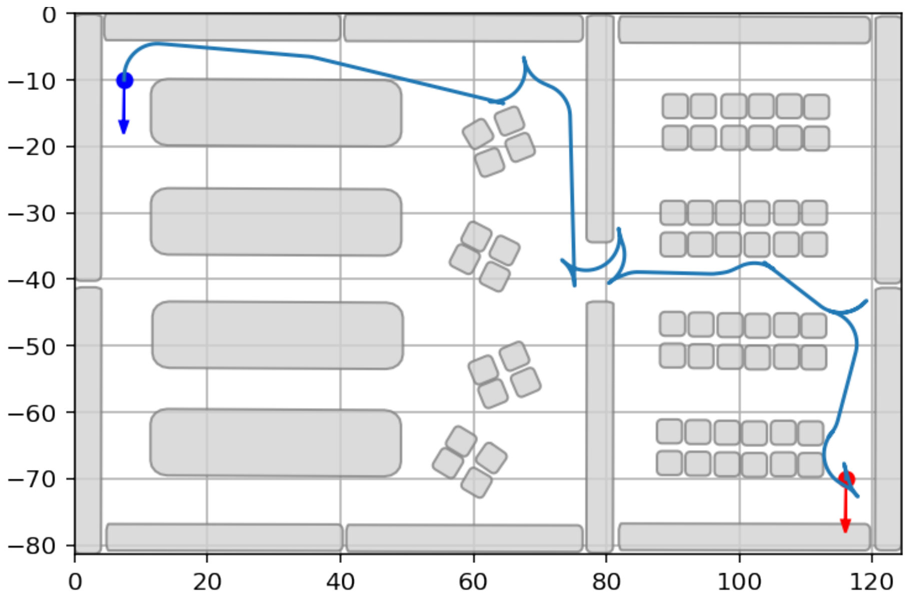
User-definable objective that is used by the sampling-based planners

- Minimize path length
- Optimize smoothness: minimizes the angle between consecutive path segments (straight line has zero smoothness) (defined in OMPL)
- Minimize curvature normalized over path length
  $\sum_i \int_{\sigma_i} \kappa\big(\dot{\sigma}_i(t)\big)\big\|\dot{p}_{\sigma_i}(t)\big\|_2 dt$ with curvature segments σ_i
  between cusps
- Maximize clearance
  (distance to nearest obstacle along the entire path)

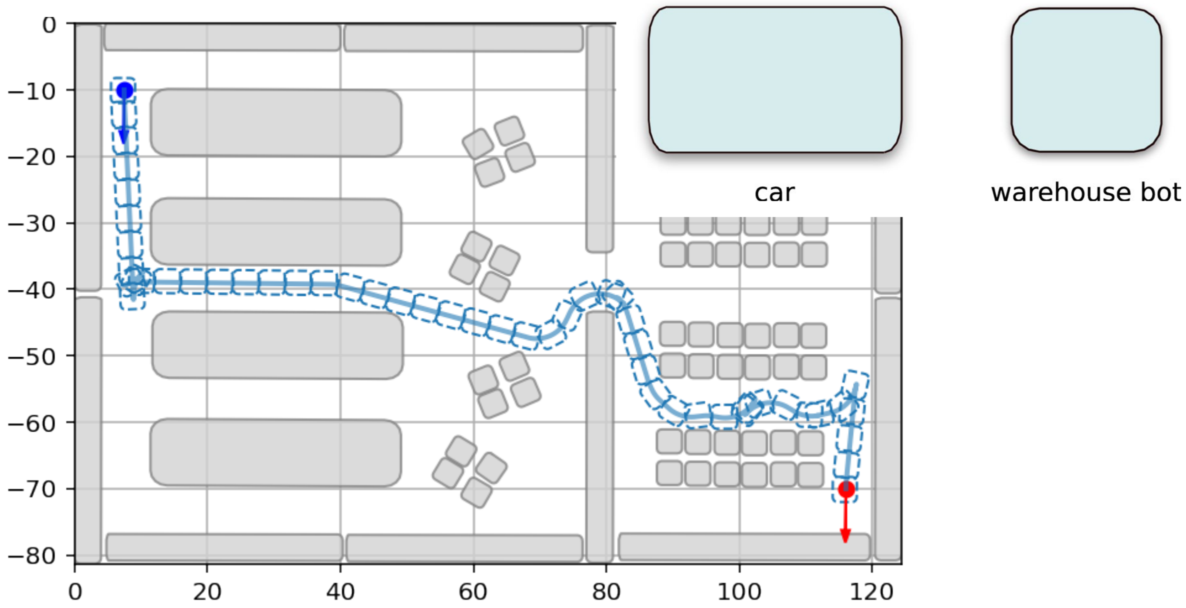
Collision Checking

Collision detection between grid, polygon environments and robot shapes represented by a point or polygon (default)

Separating Axis Theorem for CC between convex polygons



Point-based collision detection

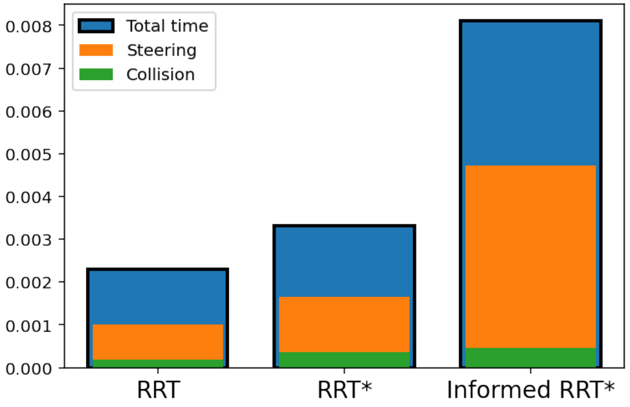


Polygon-based collision detection

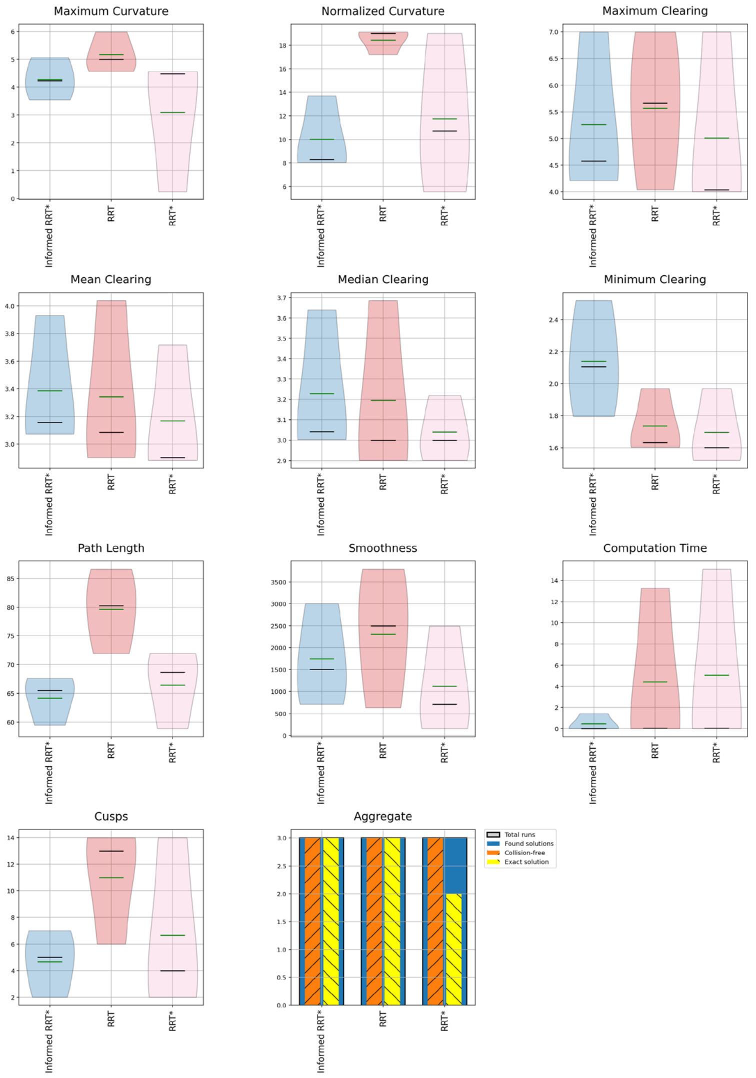
Metrics

- Path length
- Computation time (total / per planning phase)
- Clearing distance statistics (mean, median, …)
- Curvature (maximum / normalized / angle-over-length [AOL])
- Smoothness
- Number of cusps

More metrics can be added by the user



Computation time per planning phase



Various planning metrics plotted for a given benchmark

Example

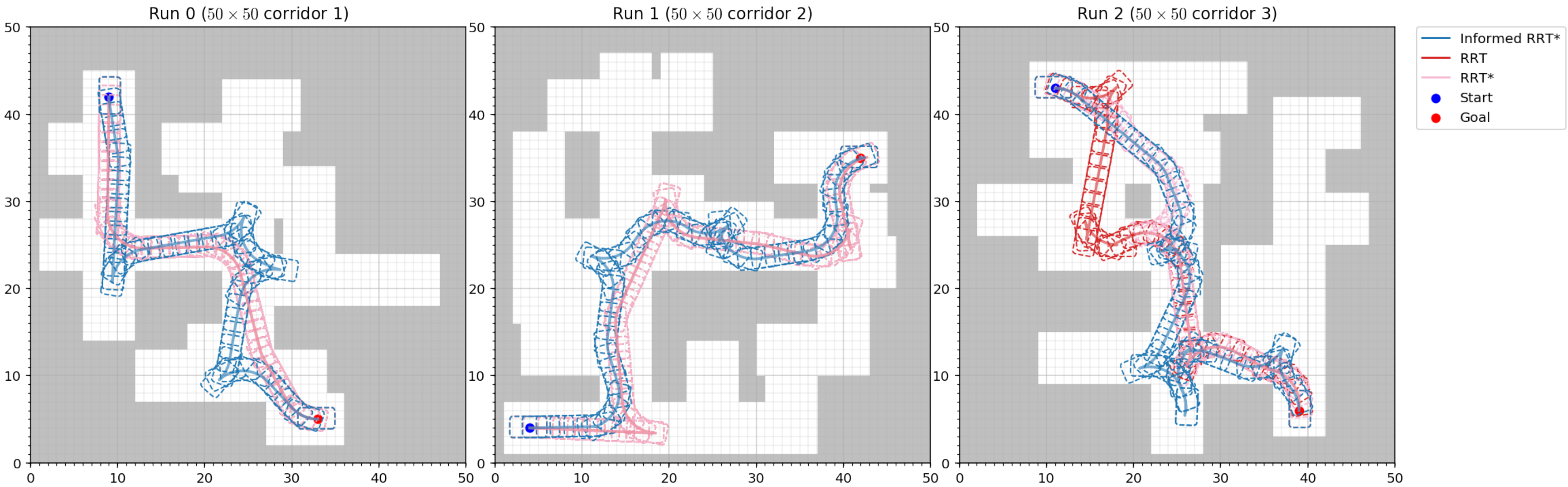
Example that shows how to use Bench-MR

Example Notebook

```python
from mpb import MPB
mpb = MPB(config_file = 'benchmark_template.json')
mpb.set_corridor_grid_env(radius = 3)
mpb.set_planners(['rrt', 'rrt_star', 'informed_rrt_star'])
mpb.set_steer_functions(['reeds_shepp'])
mpb.run(runs=3)
mpb.visualize_trajectories()
```

configuration of C++ back-end through JSON file

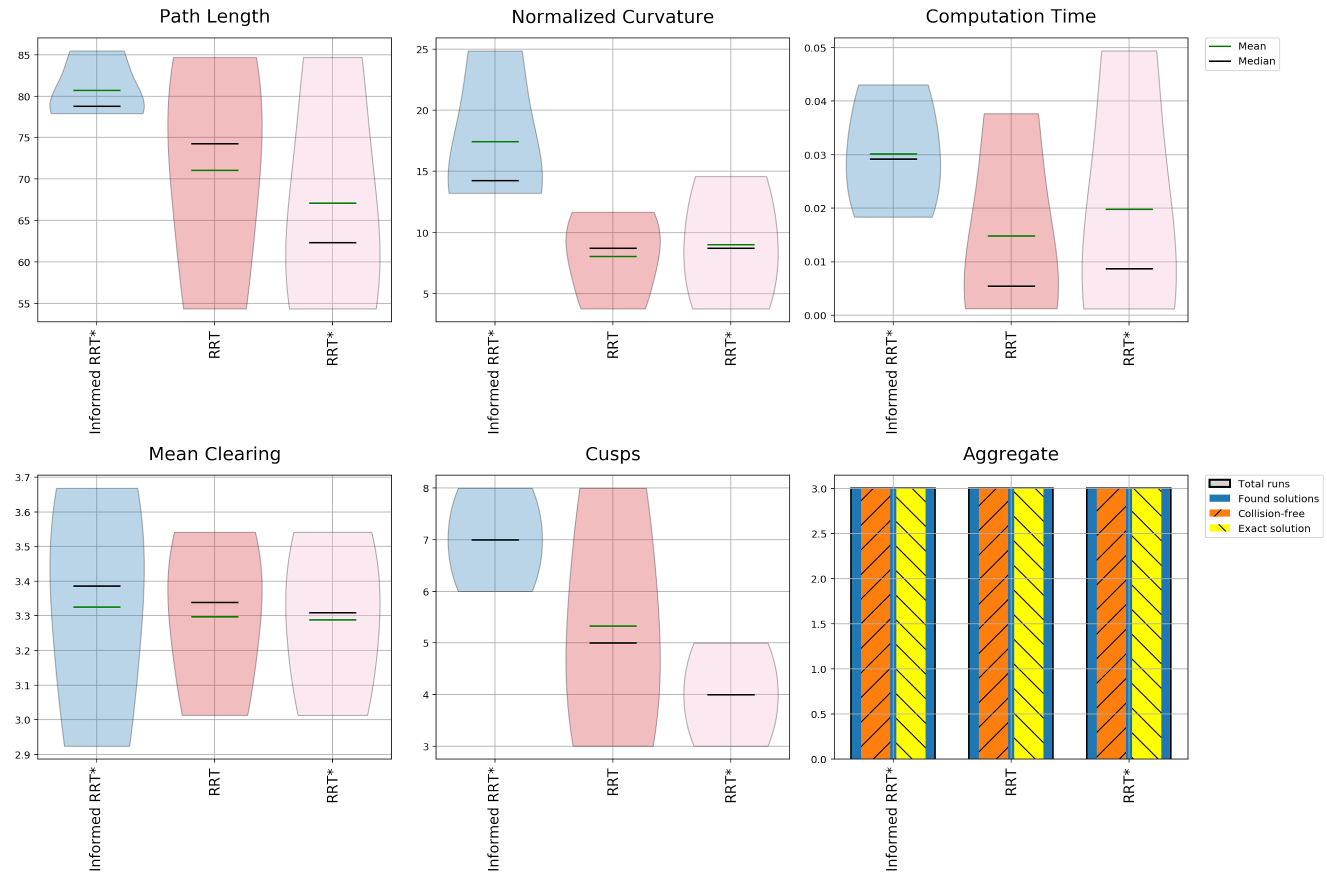
procedurally generated corridor-like environments



jupyter

Example Notebook (Cont’d)

```
mpb.plot_planner_stats()
```



Plot planning statistics for the defined set of metrics that have been evaluated

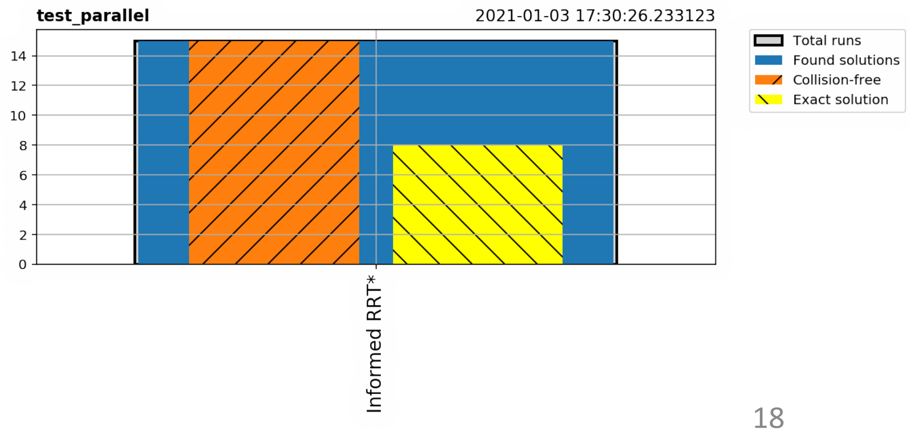
Parallelization

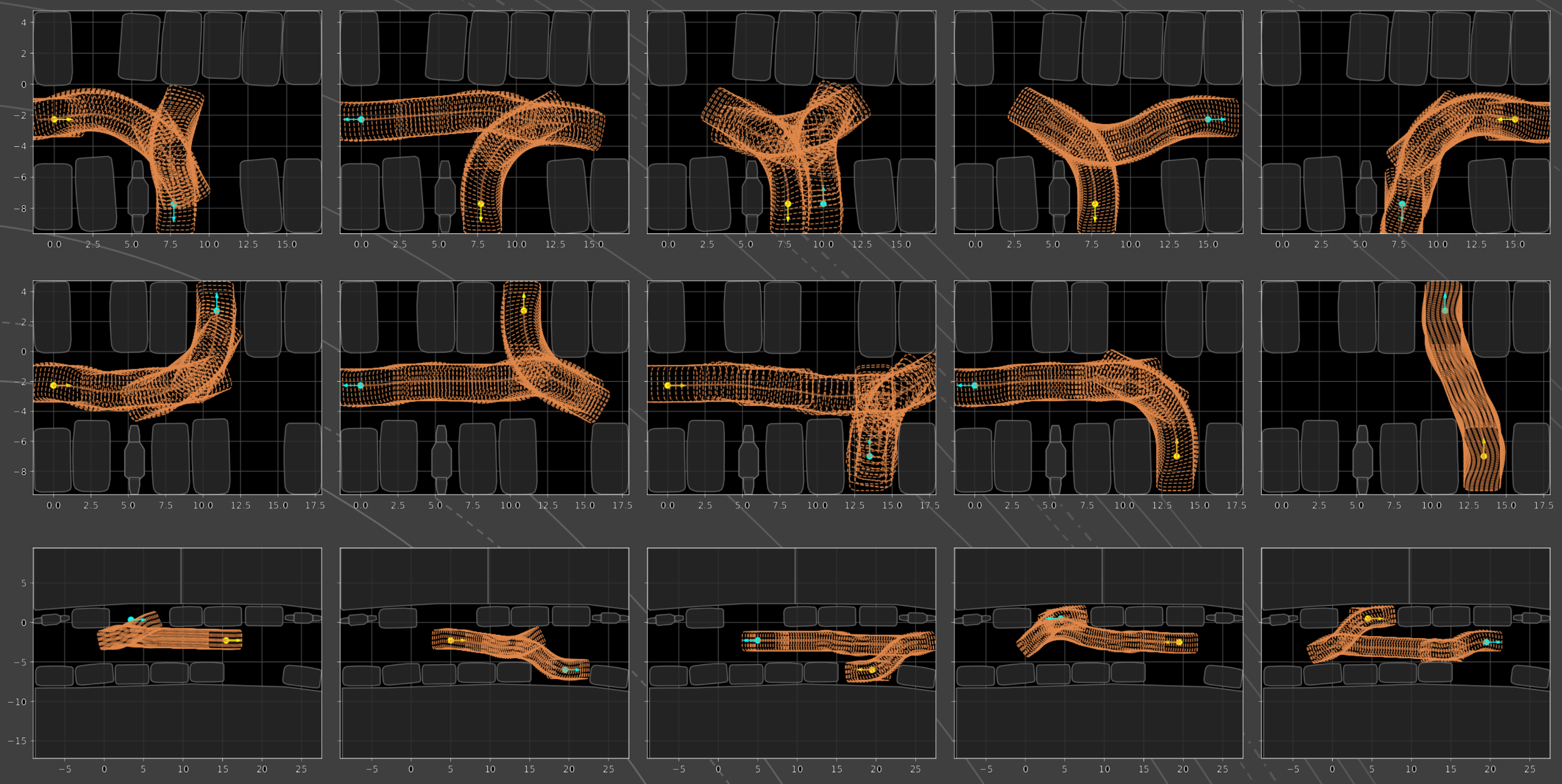
Benchmarks can be run in parallel, results are merged automatically

```
from mpb import MultipleMPB, MPB

pool = MultipleMPB()
for time in [.5, 1, 10]:
    m = MPB()
    m["max_planning_time"] = time
    m.set_corridor_grid_env()
    m.set_planners(['informed_rrt_star'])
    m.set_steer_functions(['reeds_shepp'])
    pool.benchmarks.append(m)

pool.run_parallel('test_parallel', runs=5)
```



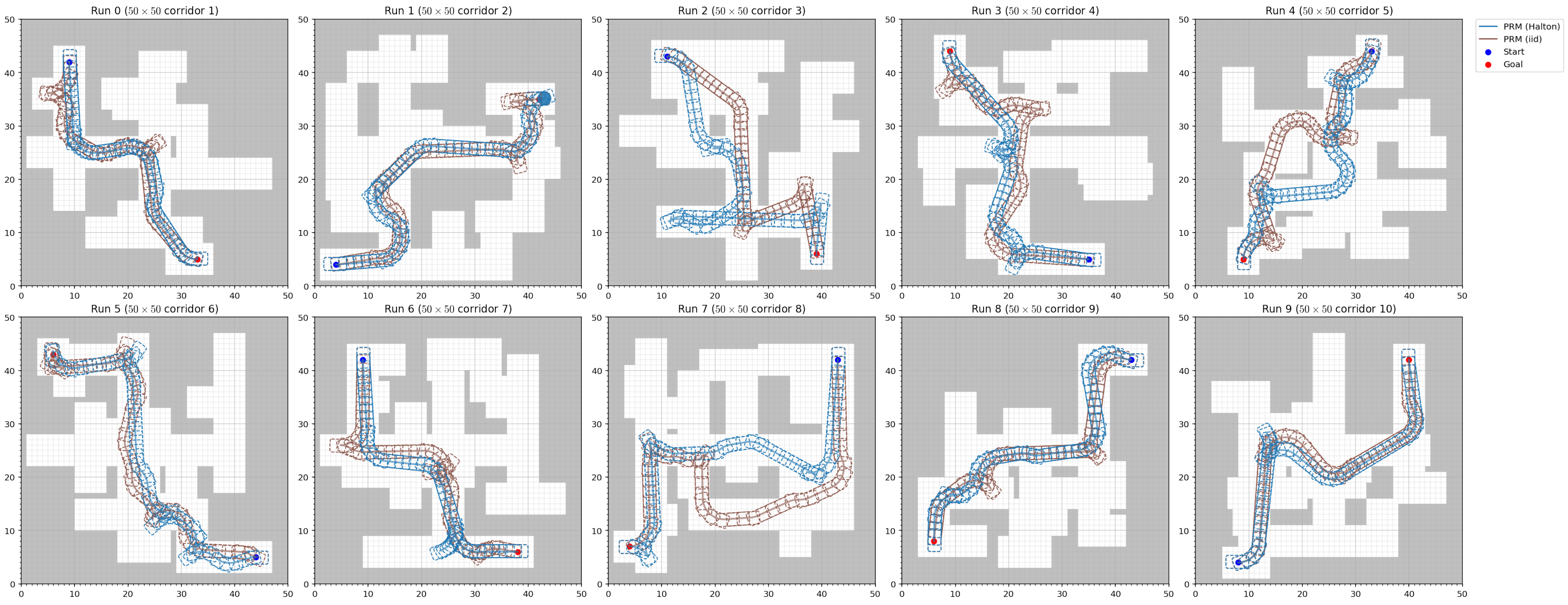


Experiments

Scenarios showcasing the features of Bench-MR

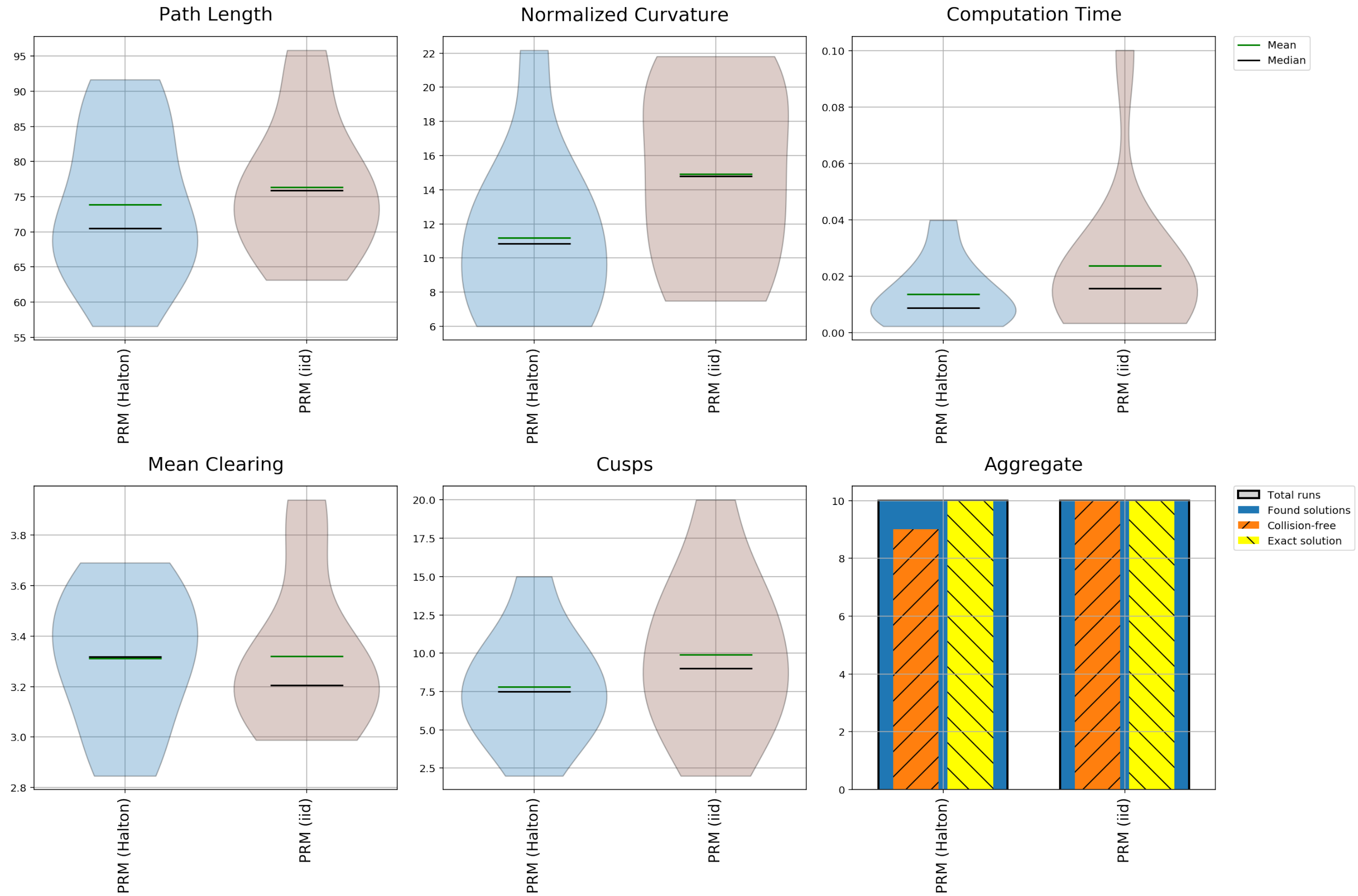
Which sampling technique is preferrable?

Benchmarking sampling-based planners using deterministic Halton sequence vs. uniform sampling



Which sampling technique is preferrable?

Planning statistics



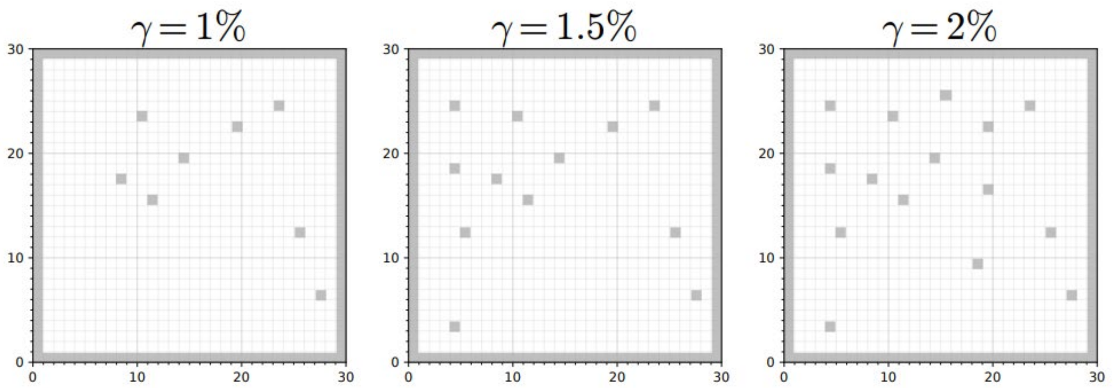
Halton sampling

Uniform random sampling

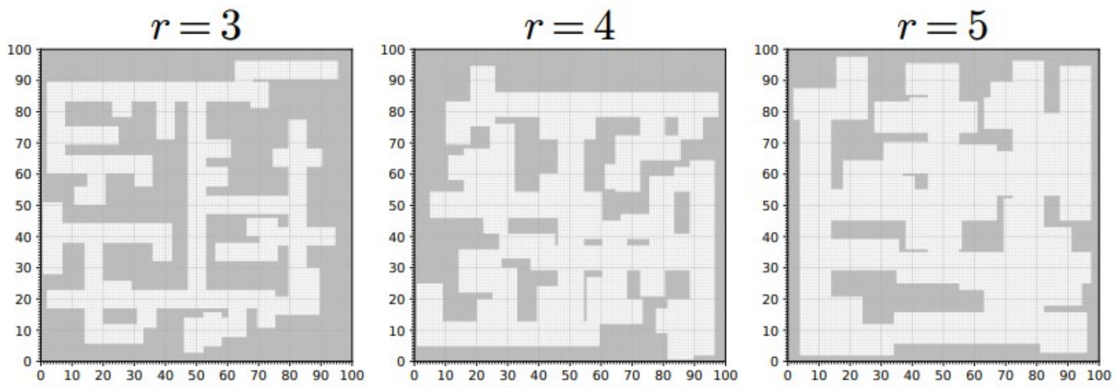
Varying Environment Complexity

Benchmarking motion planners on environments of varying complexity

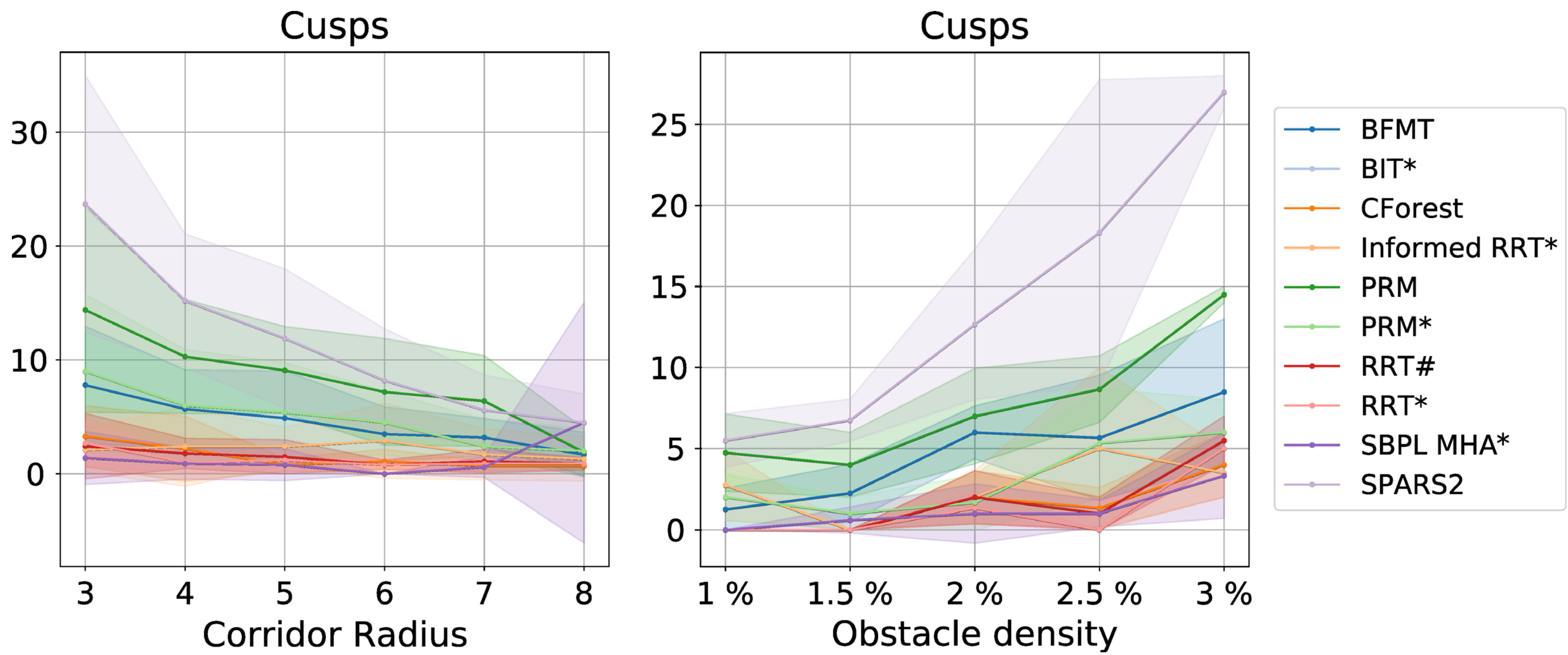
Varying obstacles density



Varying corridor size



Varying Environment Complexity



- Reeds-Shepp extend function with a computation time limit of 15 seconds each
- Procedurally generated grids with 100x100 cells

Asymptotically Optimal Planners vs. Feasible Planners with Smoothing

Is there a benefit in using a fast, feasible planner in combination with post-smoothing over asymptotically optimal motion planners?

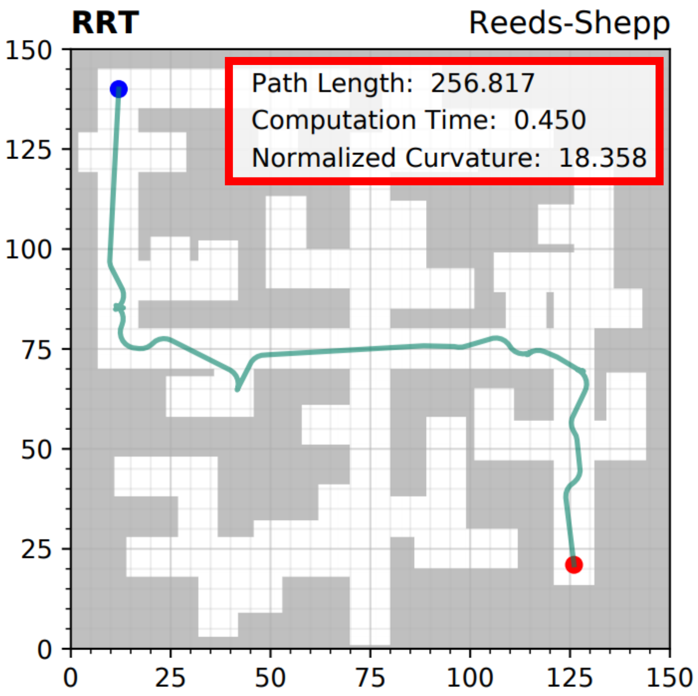
Comparison between

- Feasible planners (RRT, EST, SBL, STRIDE) + post-smoothing algorithms
- Asymptotically (near) optimal planners (RRT*, Informed RRT*, SORRT*, PRM*, CForest, BIT*, SPARS)

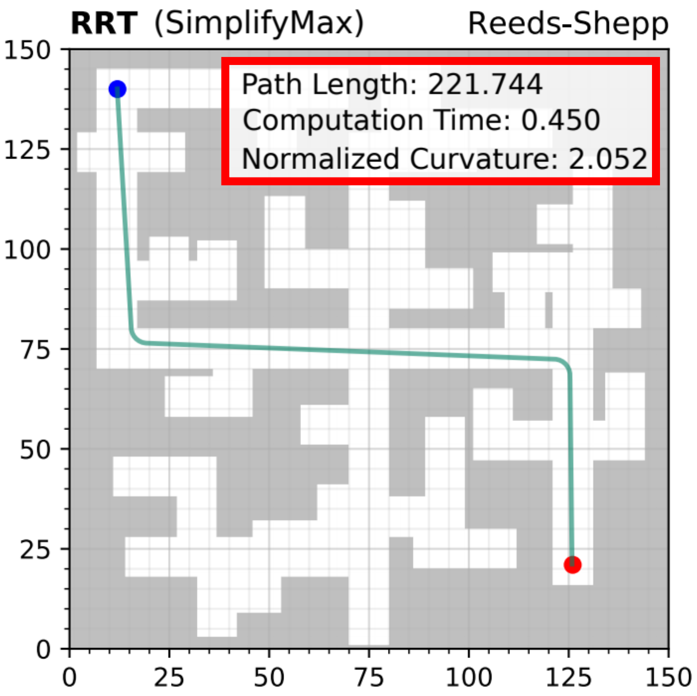
Metrics: path length and normalized curvature

Asymptotically Optimal Planners vs. Feasible Planners with Smoothing

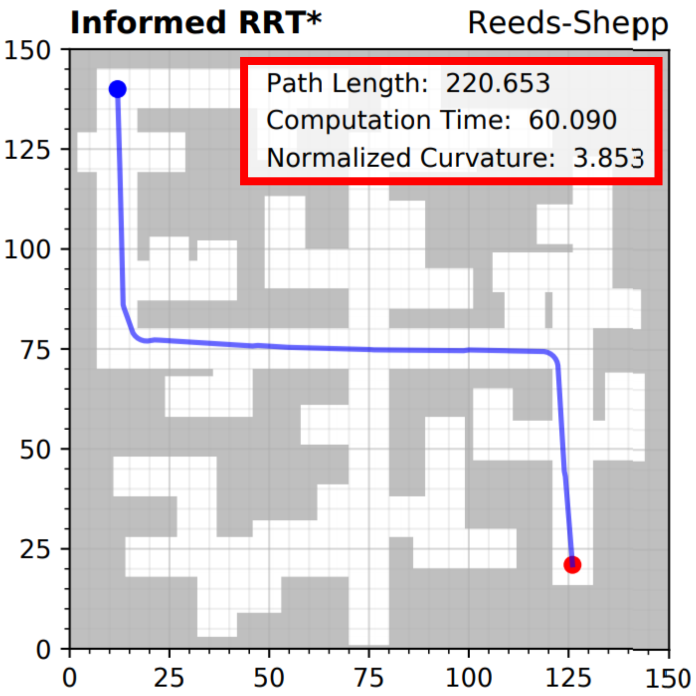
Example trajectories



Feasible planner

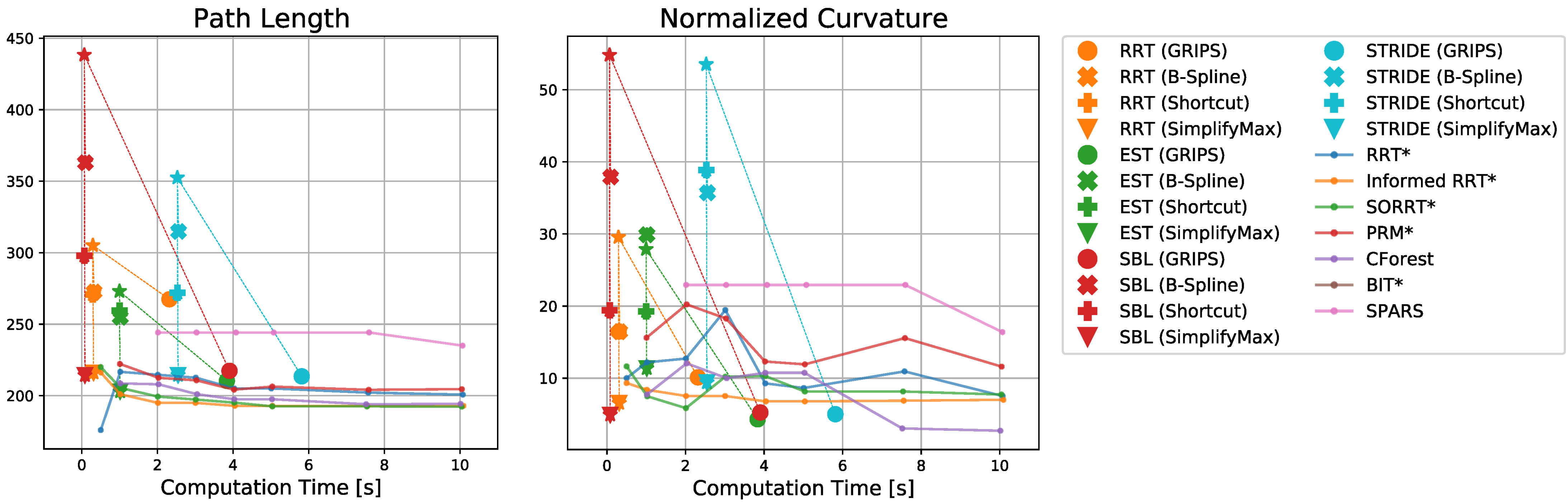


Feasible planner + post-smoothing



Asymptotically optimal planner

Asymptotically Optimal Planners vs. Feasible Planners with Smoothing

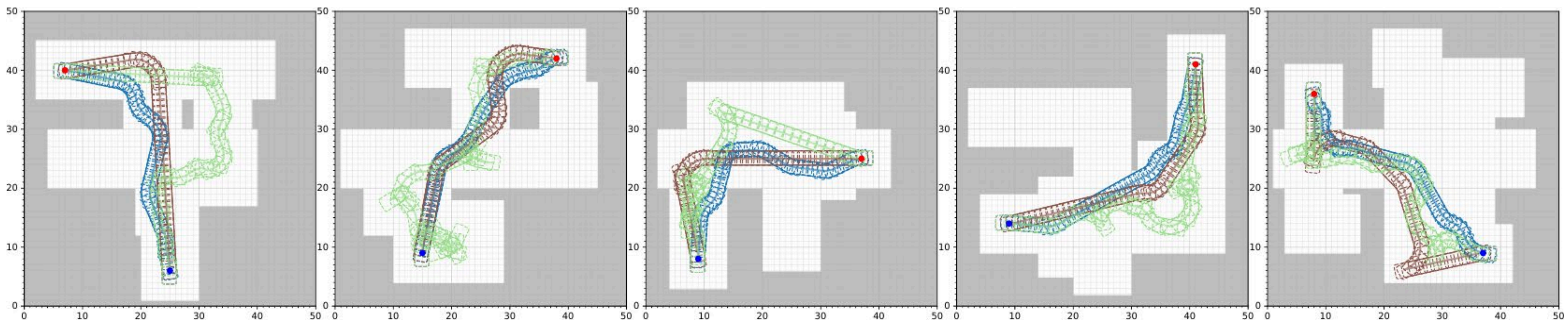


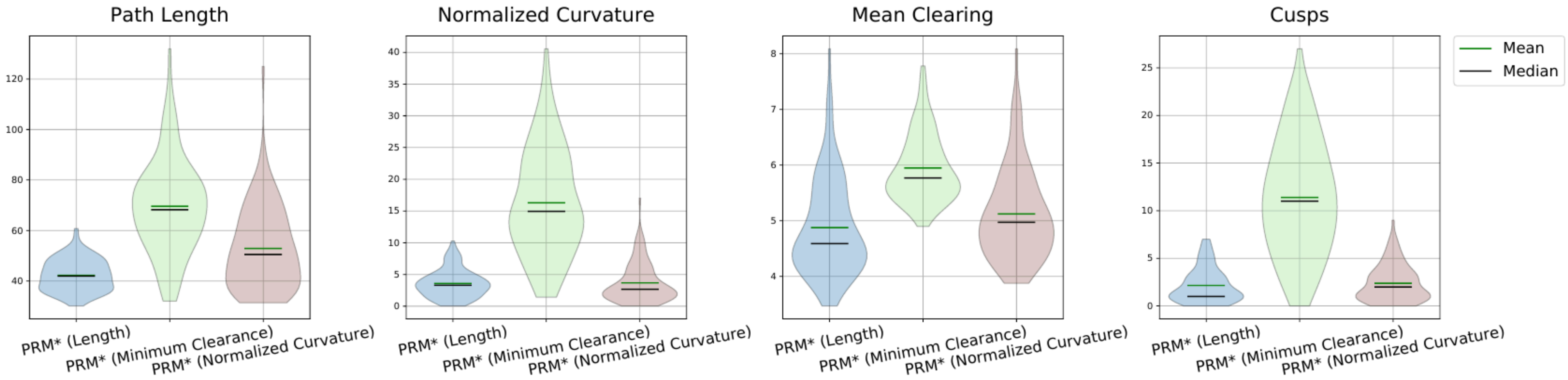
Comparison performed in random indoor-like grid-based environments of size 150×150 cells and a desired minimum corridor width of 5 cells

Optimization Objectives

- Path length
- Minimum clearance
- Normalized curvature

How do the paths differ given different optimization criteria?





Bench-MR: A Motion Planning Benchmark for Wheeled Mobile Robots

Eric Heiden[1], Luigi Palmieri[2], Leonard Bruns[3], Kai O. Arras[2], Gaurav S. Sukhatme[1], Sven Koenig[1]

[1] Dept. of Computer Science, University of Southern California, USA
[2] Robert Bosch GmbH, Corporate Research, Germany
[3] Division of Robotics, Perception and Learning (RPL), KTH, Sweden

https://robot-motion.github.io/bench-mr

USC University of Southern California

BOSCH

KTH VETENSKAP OCH KONST

IEEE 2021 ICRA
May 30 to June 5, 2021
Xi'an China