The MoveIt Benchmark Suite for Whole-Stack Planner Evaluation

Michael Görner*[1], David Pivin*[2], François Michaud[2], Jianwei Zhang[1]
Email goerner@informatik.uni-hamburg.de, david.pivin@usherbrooke.ca

***Abstract*— Whereas the performance of motion planning algorithms can be analyzed theoretically, moving toward applications based on such algorithms requires computational evaluation on likely scenarios to choose variants that perform sufficiently well on the concrete tasks and tune relevant hyperparameters. Additionally, integrated planning systems rely on various implementations for sub-features, such as inverse kinematics solving or collision checking, which have a significant impact on the overall system performance. As a result of this complexity, seemingly benign changes to the code base can entail tremendous changes in overall performance. The MoveIt Benchmark Suite (MBS) constitutes a unified benchmarking framework to evaluate the performance of the whole system stack on unit as well as integration level to compare configurations and isolate performance changes over time.**

I. INTRODUCTION

With the development and maintenance of a multi-purpose planning system, complexity grows along multiple dimensions. Among them are techniques and advances in individual components, but also advanced interplay between the subcomponents. As a result, regression testing and quality assurance, such as continuous integration tests, test coverage analysis, and performance benchmarks constitute major pillars of the development process.

We propose a system specifically targeted at the MoveIt framework [1] to facilitate community-supported development and assess performance-related changes. Many examples spanning years of maintenance can illustrate the need for such a system as behavior can regress unnoticed[1], subtle contributions can expose systematic biases with severe behavioral implications[2], and user-based performance evaluations often tend to be biased towards specific use-cases[3].

As a multi-component motion planning framework, its requirements on benchmarks fall into a gap between the idea of microbenchmarks to time individual code segments, such as supported prominently by the Benchmark framework [2], and motion planner benchmark systems such as Robowflex [3], which provide tooling and metrics to evaluate not just internal performance of planners, such as their runtime and memory use, but also external performance, measured in functions of the generated trajectories.

We provide a benchmark suite tailored to the needs of the MoveIt project that provides a common ground for unit-level benchmarks such as collision checking or inverse kinematics (IK), as well as high-level integration tests with varying scenarios. The MoveIt Benchmark Suite (MBS) builds in large parts on the Robowflex system [3], extending the scope for framework-relevant insights. The project is available at https://github.com/captain-yoshi/moveit_benchmark_suite.

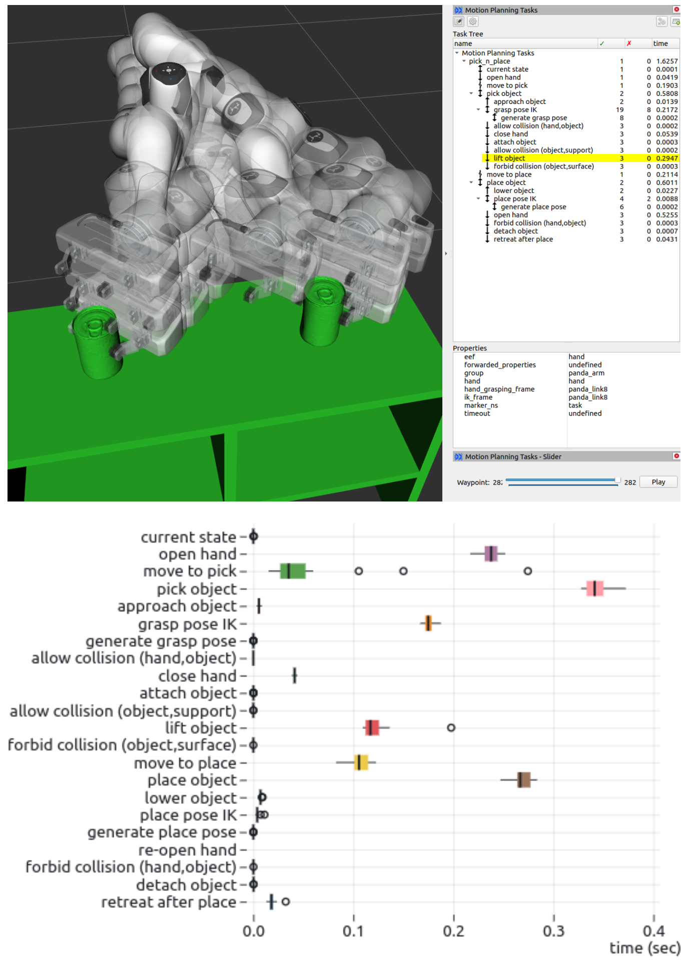

Fig. 1. MoveIt Task Constructor [4] integration benchmark with statistics measured per stage. The benchmark can capture unit-level statistics, such as IK performance, as part of an advanced integrated task such as Pick&Place.

II. BENCHMARK CATEGORIES

MBS benchmarks comprise a combination of 1) robot descriptions, 2) environments, and 3) meta parameters, such as repetition count per trial. Additionally, each benchmark type supports a set of parameters related to the concrete type.

*These authors contributed equally.
[1] Universität Hamburg, Germany.
[2] Université de Sherbrooke, Canada.

[1]moveit#197
[2]moveit#3119
[3]moveit#2698

A noteworthy extension over other systems includes the ability to describe environment scenes through multiple mechanisms, including urdf/xacro representations and clutter-generators with parameterized expected collision state.

The output metrics of each benchmark type vary as well but always include meta-data such as machine information, versions of relevant package dependencies, or git commits where available. Thus, benchmark runs can also be easily filtered/plotted/compared together across system configurations and dependency versions. As user contributions originate from various system configurations, this represents an essential capability.

A. MTC Tasks

The MoveIt Task Constructor (MTC) [4] provides mechanisms and structure to describe task-level plans and solve them in computationally independent steps using multiple backends. This explicitly modular structure makes it an ideal candidate to benchmark individual components, such as IK generators, different motion planners, and data structure operations in a complete and realistic scenario. At the same time, it allows a benchmark to directly access the effect of different configurations on extended manipulation planning. Figure 1 illustrates the traditional Pick&Place task for a can and associated benchmark statistics for the component-wise computation time.

Each stage provides general metrics for computation time and success/failure but can also report individual statistics (described in the following sections). The overall task also provides metrics for solution count, cost, and others.

B. Motion Planning Pipelines

This type of benchmark refers to the regular evaluation of motion planner performance as targeted specifically by existing systems such as Robowflex [3], and PlannerArena [5]. Parameters include algorithm-specific options, timeouts, collision detectors, IK solvers, and post-processing pipelines. Specifically, the type of goal and path constraints passed to the planner also play an essential role and can drastically affect benchmark coverage and behavior. Whereas some optimization-based trajectory solvers, such as TrajOpt [6], can handle arbitrary differentiable cost functions, many planners discriminate pose constraints from other constraint types in order to pass them to fast special-purpose IK solvers. Such constraint handling invokes very different mechanisms in the background and it is reasonable to benchmark them separately. Related regressions in the MoveIt code base were recently validated through added coverage in MBS[4].

Lastly, a variable specific to frameworks and not relevant for plain algorithm profiling is the choice of entry point for planning. Depending on the subsystems used, MoveIt provides different C++ and ROS interfaces for planning, as well as different interfaces for other components. MBS explicitly adds the possibility to benchmark planning over different entry points.

[4]moveit#3119

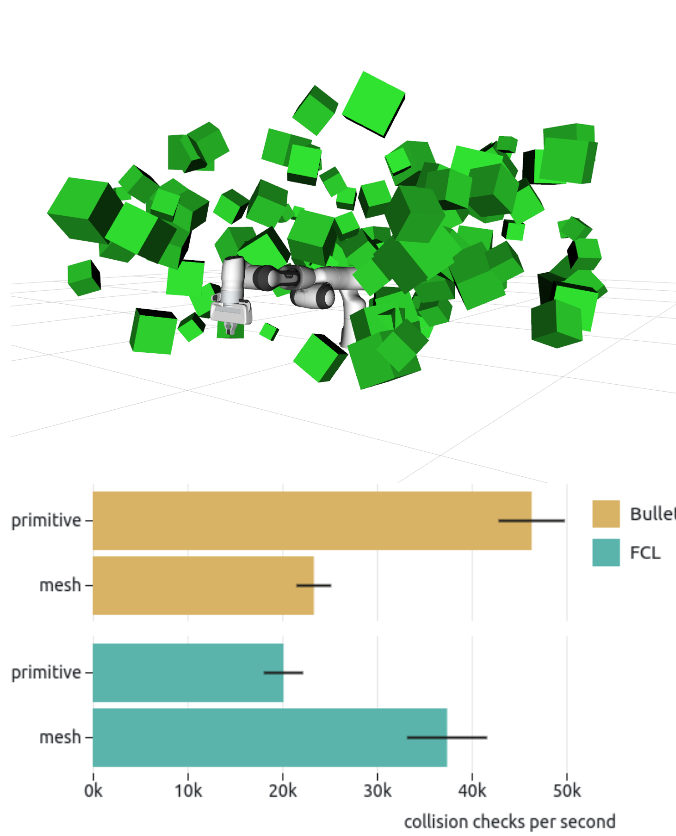

Fig. 2. Cluttered benchmark scene with 100 boxes without collisions around a Panda robot: (top) Randomized collision checking benchmark scene with cluttered boxes that do not collide with the robot. The scene can be described through box primitives or triangulated mesh representations of the boxes; (bottom) Performance of MoveIt FCL (0.6.1) and Bullet (2.88) integration w.r.t. both representations.

Various metrics are established to characterize path and motion planners, including but not limited to *time to first solution*, *memory footprint*, *path length* in joint and Cartesian space, *clearance to obstacles*, and *path smoothness*. As MBS borrows heavily from Robowflex in this benchmark category, supported metrics can be imported from there.

C. Collisions

Collision detection is a critical component of sampling-based planning and in most motion planning applications, the majority of computation time is spent on it [7]. Although the predominant Gilbert, Johnson, and Keerthi's (GJK) algorithm was published decades ago [8], impressive improvements to it are still published [9]. Additionally, the exact integration of each algorithm into the planner plays a crucial role as less efficient creation of data structures and caching behavior can hamper runtime performance.

MBS benchmarks for collision checking expose the existing MoveIt interface parameters, including binary detection and minimal distance computation, as well as the number of required collision pairs and contact points. As cluttered randomized scenes can be generated with a known number of collisions, the parameters allow for separate benchmarks that can force more or less effective use of broad-phase collision checking, accessing separate aspects of the collision checker. The benchmarks generate metrics for computation times (measured in checks per second) but also report contact counts in order to validate correctness and completeness.

Figure 2 presents a cluttered benchmark scene with 100 boxes without collisions around a Panda robot. An example benchmark configuration compares the exact scene using a primitive and a mesh representation for the oriented box collision objects. As no collision exists, a checker has to perform narrow-phase collision checks for each object inside the bounding volume hierarchy of the robot. Using MoveIt's Bullet integration, the primitive box representation (parameterized as pose and three scalars) can be queried significantly faster than the box meshes with an approximate factor of ∼2. Using FCL (Flexible Collision Library) on the other hand, the mesh representation strictly outperforms the primitive representation. This result is highly unexpected and currently being investigated[5].

D. Hyperparameters

Motion Planner performance depends significantly on hyperparameters and previous research proposed frameworks to find suitable parameter sets across problem classes [10]. Using MBS, such optimization can be considered across the whole stack. However, as the repeated evaluation of more complex benchmarks can require excessive resources, the scope of such an investigation must be limited.

As an example, Fig. 3 presents the convex mesh approximation of a UR3 robot generated from detailed meshes. As convex meshes can enable vital optimizations in collision-checking algorithms, it is often preferable to approximate mesh geometry through convex meshes at the cost of lost details and overapproximation. Using the volumetric hierarchical approximate convex decomposition approach [11], each link mesh can automatically be approximated through a varying maximum number of convex shapes. Some resulting representations are shown in the top part of the figure. These representations can be readily added to arbitrary MBS benchmarks, e.g., a reach-to-grasp task based on the *Box and Blocks Test* [12] and evaluate recorded metrics. As expected, the average planning time increases as the number of mesh faces in the reconstructions goes up, with an effective speedup of factor 2 between the detailed mesh approximation and the coarsest approximation. Depending on the relevant task, these results could be readily evaluated against opposing metrics, such as the number of reachable objects inside the box to select a task-specific trade-off.

III. Summary

The MoveIt Benchmark Suite implements a benchmark system that aims at framework development. The system is modular through a plugin mechanism and well-specified entry points in the form of `roslaunch` files. It features microbenchmark options for individual code paths, as well as motion and task-level motion planning benchmarks. The

[5] moveit#3202

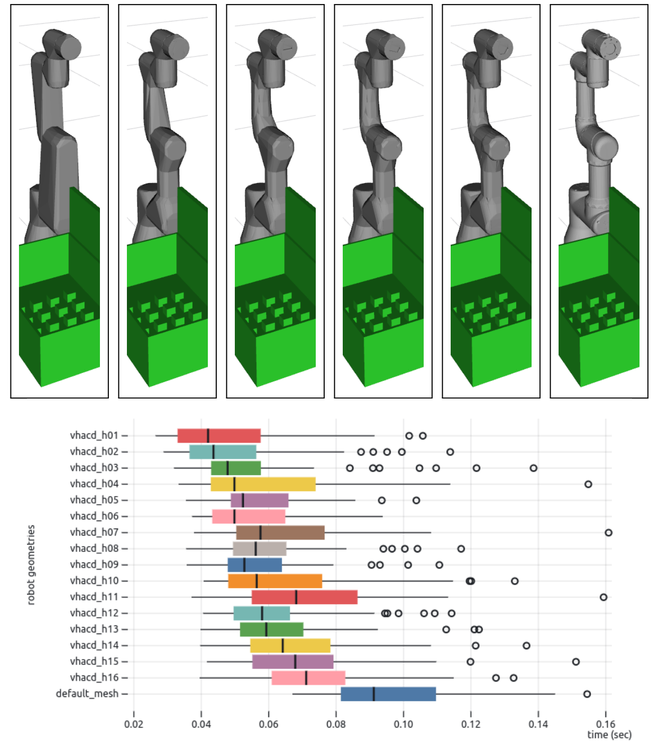

Fig. 3. Convex mesh approximation of a UR3: (top) Increasingly detailed convex decompositions and original meshes of a UR3 robot in an established *Box and Blocks Test* scenario [12]; (bottom) Planning times for a reaching motion planning request into the box. As expected, average planning times increase with more complex collision geometry.

combination allows to point out regressions and improvements on different levels of complexity to support maintenance. In future work, we plan to improve the reproducibility of recorded benchmarks. As metadata includes versions of dependencies, in principle, it is possible to generate containerized environments in a partially automated fashion to replicate benchmark runs across different hardware. It is planned to expand the set of benchmarks through plugins over time to increase coverage and help evaluate various components of the system.

IV. Acknowledgement

This project was made possible as a part of Google Summer of Code 2021. Additionally, it is partially funded by the German Research Foundation (DFG), the National Science Foundation of China in project Crossmodal Learning, TRR-169, and NSERC (Natural Sciences and Engineering Research Council) of Canada, CoRoM (Collaborative Robotics for Manufacturing) CREATE Program (# 498011-2017).

References

[1] David Coleman et al. "Reducing the barrier to entry of complex robotic software: A MoveIt! case study". In: *Journal of Software Engineering for Robotics* 5.1 (May 2014), pp. 3–16. URL: http://moveit.ros.org.

[2] Various. *Benchmark: a microbenchmark support library*. 2022. URL: `https://github.com/google/benchmark` (visited on 09/23/2022).

[3] Zachary Kingston and Lydia E. Kavraki. "Robowflex: Robot motion planning with MoveIt made easy". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Oct. 2022.

[4] Michael Görner et al. "MoveIt! Task constructor for task-level motion planning". In: *IEEE International Conference on Robotics and Automation*. 2019, pp. 190–196. DOI: `10.1109/ICRA.2019.8793898`.

[5] Mark Moll, Ioan A. Şucan, and Lydia E. Kavraki. "Benchmarking motion planning algorithms: An extensible infrastructure for analysis and visualization". In: *IEEE Robotics & Automation Magazine* 22.3 (Sept. 2015), pp. 96–102. DOI: `10.1109/MRA.2015.2448276`.

[6] John Schulman et al. "Motion planning with sequential convex optimization and convex collision checking". In: *The International Journal of Robotics Research* 33.9 (2014), pp. 1251–1270. DOI: `10.1177/0278364914528132`. eprint: `https://doi.org/10.1177/0278364914528132`. URL: `https://doi.org/10.1177/0278364914528132`.

[7] Steven M. LaValle. *Planning Algorithms*. Available at http://planning.cs.uiuc.edu/. Cambridge, U.K.: Cambridge University Press, 2006, p. 209.

[8] Elmer G. Gilbert, Daniel W. Johnson, and Sathiya S. Keerthi. "A fast procedure for computing the distance between complex objects in three-dimensional space". In: *IEEE Journal on Robotics and Automation* 4.2 (1988), pp. 193–203. DOI: `10.1109/56.2083`.

[9] Louis Montaut et al. "Collision detection accelerated: An optimization perspective". In: *Proceedings of Robotics: Science and Systems*. New York City, NY, USA, June 2022. DOI: `10.15607/RSS.2022.XVIII.039`.

[10] Mark Moll et al. "HyperPlan: A framework for motion planning algorithm selection and parameter optimization". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2021, pp. 2511–2518.

[11] Khaled Mamou. "Volumetric hierarchical approximate convex decomposition". In: *Game Engine Gems 3*. AK Peters, 2016, pp. 141–158.

[12] Andrew S. Morgan et al. "Benchmarking cluttered robot pick-and-place manipulation with the box and blocks test". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 454–461. DOI: `10.1109/LRA.2019.2961053`.