CoBRA: A Composable Benchmark for Robotics Applications

Matthias Mayer, Jonathan Külz, and Matthias Althoff

***Abstract*— Selecting an optimal robot and configuring it for a given task is currently mostly done by human expertise or trial and error. To evaluate the automatic selection and adaptation of robots to specific tasks, we introduce a benchmark suite encompassing a common format for robots, environments, and task descriptions. Our benchmark suite is especially useful for modular robots, where the multitude of robots that can be assembled creates a host of additional parameters to optimize. The benchmark suite defines these optimization problems and facilitates the comparison of solution algorithms. All benchmarks are accessible through cobra.cps.cit.tum.de, a website to conveniently share, reference, and compare solutions.**

I. Introduction

Benchmarking has supported significant progress in computer science research of computer vision [1], [2], robotic skills, such as grasping [3], or motion planning [4], [5]. Recent advances in Modular Reconfigurable Robots (MRRs) have shown that they can be more efficient in automating industrial processes [6]–[8]. Comparing these approaches was done by re-implementation by each author, making the comparisons hard to trust.

Therefore, we propose CoBRA, a **Co**mposable **B**enchmark for **R**obotics **A**pplications, which includes standardized descriptions for many MRR models, typical industrial robot tasks, and robotic deployments that solve these tasks. Each of these contributions is hosted on the central website cobra.cps.cit.tum.de where they are available for anyone.

II. Method

CoBRA considers the optimization problem for (modular) robots. Its goal is to minimize a cost function J_C by optimizing a module composition M, base pose $\mathbf{B}$, and joint trajectory $\vec{x}$:

$$[M^*, \mathbf{B}^*, \vec{x}^*] = \underset{M,\mathbf{B},\vec{x}}{\arg\min}\, J_C(\vec{x}(t), \mathbf{B}, M) \qquad (1)$$

By assuming different parts fixed, we include subproblems such as robot base placement optimization (M fixed), or robot path planning (M and $\mathbf{B}$ fixed). We provide multiple cost functions for the robot complexity, e.g., number of joints, trajectory costs, such as mechanical energy, and degree of task fulfillment, e.g., number of solved goals.

All authors are with the Technical University of Munich, TUM School of Computation, Information and Technology, Chair of Robotics, Artificial Intelligence and Real-time Systems, Boltzmannstraße 3, 85748, Garching, Germany. {matthias.mayer, jonathan.kuelz, althoff}@tum.de

A. Robot

In CoBRA we aim to formalize any conceivable rigid Modular Reconfigurable Robot (MRR), which also includes any manipulator arm. A serial robot M is constructed as a vector of modules $M = (m_1, ..., m_N)$, where each module m_i is in a set of available modules $\mathcal{R}$. A more powerful description, e.g., for parallel robots, is also provided[1].

Each module describes a combination of bodies connected via joints, extending the formalization in [6] to modules with more than one joint. Bodies describe inertia, collision information, and provide connectors to other bodies. Connectors are an extension of [9], adding size and type parameters to the gender-based connections. Every joint models the relative movement of two bodies and can encode dynamics with inertia from a geared drive-train, and friction.

B. Task

A task Θ in CoBRA describes what a robot should do and the environment it should be done in. It gives a set of static obstacles $\mathcal{O}$, constraints to obey $\mathcal{C}$, and goals to achieve $\mathcal{G}$. Obstacles are described by geometric primitives or triangulated meshes. The benchmark suite already includes a set of common machines and a few 3D scans of real-world machine-shops.

Both constraints and goals are defined as functions, $c(\vec{x}(t), t, \mathbf{B}, M) \in \mathcal{C}$ and $g(\vec{x}(t), t, \mathbf{B}, M) \in \mathcal{G}$, respectively. A constraint is fulfilled if $\forall t : c(\vec{x}(t), t, \mathbf{B}, M) \leq 0$. CoBRA contains common robotic constraints, such as joint limits, freedom of collisions, and limitations on the orientation of the end-effector. Constraints on the base pose of the robot allow to simplify or remove the optimization of the base-pose $\mathbf{B}$. A solution to all goals in $\mathcal{G}$ can be enforced, including an optional order of goals.

A goal is fulfilled if its termination condition g evaluates to true. We provide goal primitives for reaching (and stopping) at specific task space poses, returning to previous states for cyclic movements, pausing, trajectory following, or leaving specific areas, such as the workspace of a machine. We want to especially highlight our formalism for arbitrarily tolerated poses in the workspace. We consider any pose $\mathbf{T} \in SE(3)$ as valid for a desired pose $\mathbf{T}_d \in SE(3)$ if $S(\mathbf{T}_d^{-1}\mathbf{T}) \in \Gamma(\mathbf{T}_d)$. Here, S is an arbitrary combination of projection functions, e.g., to the Cartesian position x, y, z, or roll, pitch, yaw angles, and Γ contains intervals for each projection to be considered valid.

[1]https://cobra.cps.cit.tum.de/robot_description

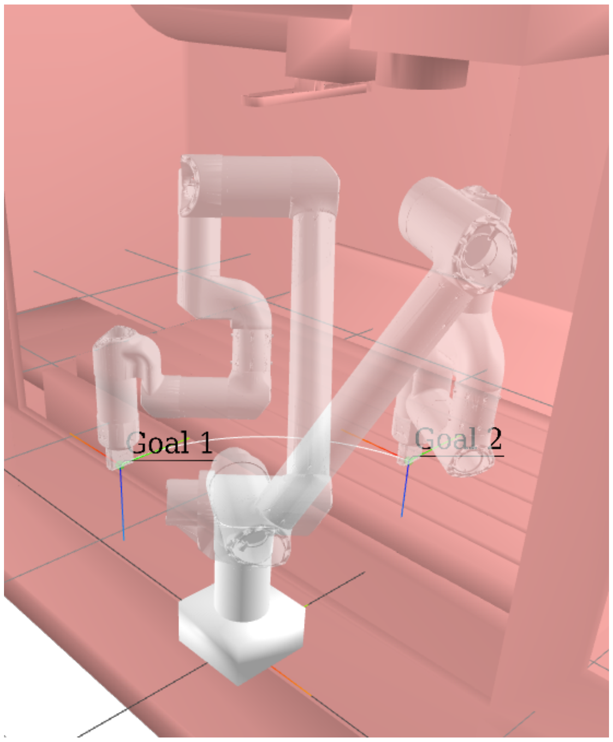

Fig. 1. A solution path for task simple/PTP_3 with cost $[(J_T|1)]$. The solving robot was constructed from the set PROMODULAR and has the module order $M = [59, 3, 55, 3, 40, 4, 38, 5, 24, 6, 54, 6, 61]$.

C. Solution

Each solution to a task Θ specifies an assembled MRR, its base pose $\mathbf{B}$, a cost function J_C, and solving trajectory $\vec{x}$ describing the joint state of the robot over time. By recombining different tasks, robot module sets, and cost functions, the benchmark suite implements composability to adapt it to evaluate different optimization problems in the area of MRRs. A specific solution can be uploaded to the benchmark suite's website for reference and comparison[2].

III. Discussion

At the moment, we include a set of 340 task descriptions in CoBRA. They are based on 20 industrial machines and objects to interact with, including five high-resolution 3D scans from a machine-shop. A set of test tasks is provided under the prefix of `simple/`[3] that show the usage of various goal types and constraints. A big batch of synthetic tasks is based on the sampling described in [10], with a grid of cubic obstacles and random desired poses in the free-space, are prefixed with `Whitman2020/`. The 3D scans and several CAD models of CNC machines have been used for task generation similar to case 2b in [7] and can be found by their prefix `Liu2020/`.

We would like to highlight a sample solution to discuss the comparison features of CoBRA. It is a simple point-to-point movement task and shown in Fig. 1. The website lists solutions by various users and can be filtered by desired cost-functions on the top. In the detailed view the solution file can be downloaded, and we show a 3D animation of the solution, as well as the calculated costs and a reported computation time to find the solution.

New users can find a thorough documentation of all of CoBRA's features and a getting started guide on our website[4]. If questions or suggestions arise they can be discussed in the forum. In addition, we provide CoBRA I/O to interface with the benchmark suite in python and timor-python [11] which provides many utilities such as a robot simulator and verification for many of the discussed CoBRA features.

IV. Conclusion

This extended abstract introduced CoBRA, a benchmark suite for Modular Reconfigurable Robot (MRR) composition optimization. CoBRA contains formal descriptions for MRRs, tasks they should fulfill, and solutions that describe how specific robot assemblies minimize a cost-function in a given task. With over 300 tasks CoBRA contains a variety of automatically generated and real-world tasks defined in 3D scans. We provide tools to interact with the benchmark, and invite the community to provide their own benchmark tasks or use it to compare their MRR optimization algorithms.

Acknowledgements

This work was supported by the EU's Horizon 2020 project CONCERT (grant no. 101016007).

References

[1] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vision*, vol. 115, no. 3, pp. 211–252, 2015.
[2] T.-Y. Lin *et al.*, "Microsoft COCO: Common Objects in Context," in *Comput. Vision*, 2014, pp. 740–755.
[3] J. Mahler *et al.*, "Learning Ambidextrous Robot Grasping Policies," *Sci. Robot.*, vol. 4, no. 26, p. eaau4984, 2019.
[4] M. Moll, I. Şucan, and L. Kavraki, "Benchmarking Motion Planning Algorithms: An Extensible Infrastructure for Analysis and Visualization," *IEEE Robot. Autom. Mag.*, vol. 22, no. 3, pp. 96–102, 2015.
[5] E. Althaus *et al.*, "Verification of linear hybrid systems with large discrete state spaces using counterexample-guided abstraction refinement," *Science of Computer Programming*, vol. 148, pp. 123–160, 2017.
[6] M. Althoff, A. Giusti, S. B. Liu, and A. Pereira, "Effortless Creation of Safe Robots from Modules through Self-Programming and Self-Verification," *Sci. Robot.*, vol. 4, no. 31, p. aaw1924, 2019.
[7] S. B. Liu and M. Althoff, "Optimizing Performance in Automation through Modular Robots," in *Proc. - IEEE Int. Conf. Robot. Autom.*, 2020, pp. 4044–4050.
[8] J. Whitman and H. Choset, "Task-Specific Manipulator Design and Trajectory Synthesis," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 301–308, 2019.
[9] M. Bordignon, K. Stoy, and U. P. Schultz, "Generalized Programming of Modular Robots through Kinematic Configurations," in *IEEE Int. Conf. Intell. Robots Syst.*, 2011, pp. 3659–3666.
[10] J. Whitman, R. Bhirangi, M. Travers, and H. Choset, "Modular Robot Design Synthesis with Deep Reinforcement Learning," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 10 418–10 425.
[11] J. Külz, M. Mayer, and M. Althoff, "Timor Python: A Toolbox for Industrial Modular Robotics," arXiv:2209.06758 [cs.RO], 2022.

[2] `https://cobra.cps.cit.tum.de/new-submission`

[3] Use this in the field "(Partial) Task Name" on `https://cobra.cps.cit.tum.de/tasks`.

[4] `https://cobra.cps.cit.tum.de/crok-documentation/tutorial`