The Journal of Open Source Software

The ARC-OPT Library for Whole-Body Control of Robotic Systems

Dennis Mronga [1] **and Frank Kirchner** [1,2]

1 German Research Center for Artificial Intelligence (DFKI), Bremen, Germany **2** University of Bremen, Bremen, Germany

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 07 March 2024
Published: unpublished

Summary

ARC-OPT (Adaptive Robot Control using Optimization) is a C++ library for Whole-Body Control (WBC) (Sentis & Khatib, 2006) of complex robotic systems, such as humanoids, quadrupedal robots, or mobile manipulators.

WBC aims to describe a robot control problem in terms of costs and constraints of a quadratic program (QP) and to design a set of feedback controllers around it, each dedicated to a specific robot tasks. In every control cycle, the QP is solved and the solution is applied to the robot's actuators. WBC is a reactive control approach, which targets redundant robots and is able to control multiple tasks simultaneously, like, e.g., grasping and balancing on a humanoid robot.

Statement of need

ARC-OPT supports the software developer in designing robot controllers by providing configuration options for different pre-defined WBC problems. In contrast to existing libraries (Prete et al., 2016),(Posa et al., 2016), ARC-OPT provides unified interfaces for different WBC problems on velocity, acceleration and torque level, as well as options to benchmark different QP solvers and rigid body dynamics libraries on these problems. Finally, it provides a WBC approach for robots with parallel kinematic loops, as described in (Mronga et al., 2022).

The Journal of Open Source Software

Description

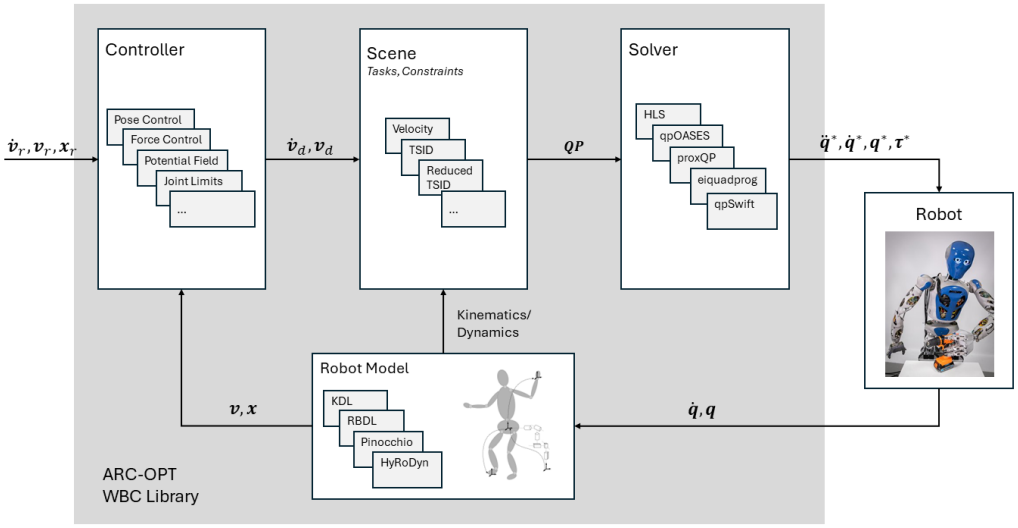

Figure 1: ARC-OPT library overview

Figure 1 shows an overview of the ARC-OPT library. ARC-OPT separates the implementation of controllers, robot model, solver, and scene, which allows a modular composition of the WBC problem:

- A **controller** implements a function in the robot's task space, e.g., for maintaining balance, avoiding an obstacle, or following a trajectory. ARC-OPT provides various controllers in joint or Cartesian space, like PD-Controllers, or repulsive potential fields.
- The **scene** sets up the QP, where the costs can be configured at runtime, and the constraints are specific for the implemented scene. Different scenes are currently implemented on velocity, acceleration and torque level.
- The **robot model** computes the kinematic and dynamic information that the scene requires to set up the QP, like Jacobians, mass-inertia matrices, and gravity terms. ARC-OPT implements various robot models based on Pinocchio (Carpentier et al., 2019), RBDL (Felis, 2016), KDL (Smits, n.d.), and Hyrodyn (Kumar et al., 2020).
- The **solver** solves the QP set up in the scene and generates the required control input for the robot joints. ARC-OPT provides various QP solvers based on open-source implementations, e.g., qpOASES (Ferreau et al., 2014), eiquadprog (Buondonno, 2021), proxQP (Bambade et al., 2022), and qpSwift (Pandala et al., 2019).

Apart from this, ARC-OPT implements various concepts typically used in WBC, like floating base dynamics, friction cone constraints, cost weighting, and task hierarchies.

Example

This example shows how to set up an acceleration/torque-level WBC. Here, the tasks are formulated in the cost function. Equations of motion, rigid contacts and joint torque limits are implemented as constraints. The decision variables are the joint accelerations $\ddot{\mathbf{q}}$, joint torques τ and contact wrenches $\mathbf{f}$. Mathematically, this can be expressed by the following QP:

$$\begin{aligned} \min_{\ddot{\mathbf{q}},\tau,\mathbf{f}} \quad & \|\mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}} - \dot{\mathbf{v}}_d\|_2 \\ \text{s.t.} \quad & \mathbf{H}\ddot{\mathbf{q}} + \mathbf{h} = \tau + \mathbf{J}_c\mathbf{f} \\ & \mathbf{J}_c\ddot{\mathbf{q}} = -\dot{\mathbf{J}}_c\dot{\mathbf{q}} \\ & \tau_m \leq \tau \leq \tau_M \end{aligned}$$

where $\mathbf{J}$ is the robot Jacobian, $\dot{\mathbf{v}}_d$ the desired task space acceleration, $\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}$ the joint positions, velocities, and accelerations $\mathbf{H}$ the mass-inertia matrix, $\mathbf{h}$ the vector of gravity and Coriolis forces, τ the robot joint torques, $\mathbf{f}$ the contact wrenches, $\mathbf{J}_c$ the contact Jacobian, and τ_m, τ_M the joint torque limits. To implement a simple Cartesian position controller for, e.g., controlling the end effector of a robot manipulator, the following PD-controller can be used to generate $\dot{\mathbf{v}}_d$:

$$\dot{\mathbf{v}}_d = \dot{\mathbf{v}}_r + \mathbf{K}_d(\mathbf{v}_r - \mathbf{v}) + \mathbf{K}_p(\mathbf{x}_r - \mathbf{x})$$

where $\mathbf{K}_p, \mathbf{K}_d$ are gain matrices, $\mathbf{x}, \mathbf{v}$ the end effector position and velocity, $\dot{\mathbf{v}}_r, \mathbf{v}_r, \mathbf{x}_r$ the reference acceleration, velocity, and position. This example is implemented for a 7 DoF KUKA iiwa robot arm in the ARC-OPT tutorials[1].

The ARC-OPT library for Whole-Body Control has been used in various scientific works (Mronga et al., 2022),(Mronga & Kirchner, 2021),(Mronga et al., 2020),(Popescu et al., 2022), and evaluated on different robots, like, e.g., the RH5 humanoid (Eßer et al., 2021) shown in Figure 2.

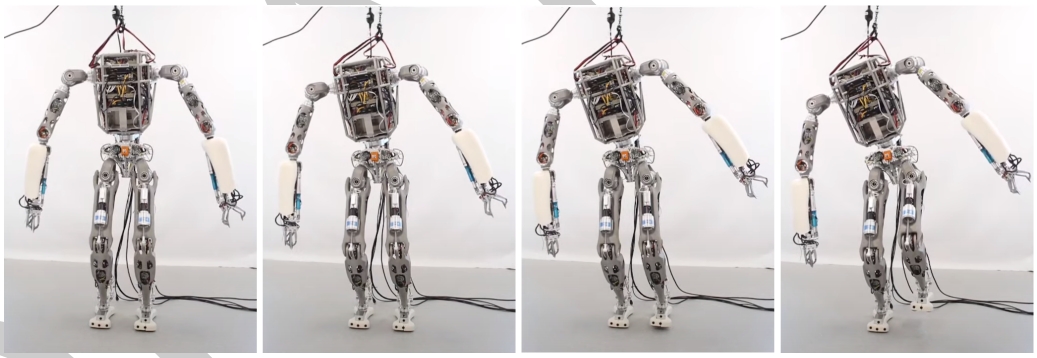

Figure 2: RH5 Humanoid robot standing on one leg using the ARC-OPT library

Acknowledgements

ARC-OPT is supported through grants from the German Federal Ministry of Education and Research (BMBF), grant numbers 01IW21002 (M-Rock project) and 01IW20004 (VeryHuman project).

References

Bambade, A., El-Kazdadi, S., Taylor, A., & Carpentier, J. (2022, June). PROX-QP: Yet another Quadratic Programming Solver for Robotics and beyond. *RSS 2022 - Robotics: Science and Systems*. https://inria.hal.science/hal-03683733

Buondonno, G. (2021). *Eiquadprog*. https://github.com/stack-of-tasks/eiquadprog

[1] https://github.com/ARC-OPT/wbc/blob/master/tutorials/kuka_iiwa/cart_pos_ctrl_dynamic.cpp

Carpentier, J., Saurel, G., Buondonno, G., Mirabel, J., Lamiraux, F., Stasse, O., & Mansard, N. (2019). The pinocchio c++ library – a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. *IEEE International Symposium on System Integrations (SII)*.

Eßer, J., Kumar, S., Peters, H., Bargsten, V., Gea, J. de, Mastalli, C., Stasse, O., & Kirchner, F. (2021). Design, analysis and control of the series-parallel hybrid RH5 humanoid robot. *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*, 400–407. https://api.semanticscholar.org/CorpusID:231709495

Felis, M. L. (2016). RBDL: An efficient rigid-body dynamics library using recursive algorithms. *Autonomous Robots*, 1–17. https://doi.org/10.1007/s10514-016-9574-0

Ferreau, H. J., Kirches, C., Potschka, A., Bock, H. G., & Diehl, M. (2014). qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, *6*(4), 327–363.

Kumar, S., Szadkowski, K. A. von, Mueller, A., & Kirchner, F. (2020). An analytical and modular software workbench for solving kinematics and dynamics of series-parallel hybrid robots. *Journal of Mechanisms and Robotics*, *12*(2). https://doi.org/10.1115/1.4045941

Mronga, D., & Kirchner, F. (2021). Learning context-adaptive task constraints for robotic manipulation. *Robotics and Autonomous Systems*, *141*, 103779. https://doi.org/https://doi.org/10.1016/j.robot.2021.103779

Mronga, D., Knobloch, T., Gea Fernández, J. de, & Kirchner, F. (2020). A constraint-based approach for human-robot collision avoidance. *Advanced Robotics*, 1–17.

Mronga, D., Kumar, S., & Kirchner, F. (2022). Whole-body control of series-parallel hybrid robots. *2022 International Conference on Robotics and Automation (ICRA)*, 228–234. https://doi.org/10.1109/ICRA46639.2022.9811616

Pandala, A. G., Ding, Y., & Park, H.-W. (2019). qpSWIFT: A real-time sparse quadratic program solver for robotic applications. *IEEE Robotics and Automation Letters*, *4*(4), 3355–3362.

Popescu, M., Mronga, D., Bergonzani, I., Kumar, S., & Kirchner, F. (2022). Experimental investigations into using motion capture state feedback for real-time control of a humanoid robot. *Sensors*, *22*(24). https://doi.org/10.3390/s22249853

Posa, M., Kuindersma, S., & Tedrake, R. (2016). Optimization and stabilization of trajectories for constrained dynamical systems. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 1366–1373. https://doi.org/10.1109/ICRA.2016.7487270

Prete, A. del, Mansard, N., Ramos Ponce, O. E., Stasse, O., & Nori, F. (2016). Implementing Torque Control with High-Ratio Gear Boxes and without Joint-Torque Sensors. *International Journal of Humanoid Robotics*, *13*(1), 1550044. https://hal.science/hal-01136936

Sentis, L., & Khatib, O. (2006). A whole-body control framework for humanoids operating in human environments. *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 2641–2648. https://doi.org/10.1109/ROBOT.2006.1642100

Smits, R. (n.d.). *KDL: Kinematics and Dynamics Library*. http://www.orocos.org/kdl