Robomorphic Computing: A Design Methodology for Domain-Specific Accelerators Parameterized by Robot Morphology

Extended Abstract

Sabrina M. Neuman†, Brian Plancher†, Thomas Bourgeat‡, Thierry Tambe†,
Srinivas Devadas‡, Vijay Janapa Reddi†

† Harvard University
‡Massachusetts Institute of Technology
sneuman@seas.harvard.edu, brian_plancher@g.harvard.edu, bthom@csail.mit.edu, ttambe@g.harvard.edu,
devadas@mit.edu, vj@eecs.harvard.edu

1. Motivation

We tackle the performance bottleneck of robot motion planning and control, addressing it with a systematic hardware design methodology that is portable across robot platforms.

Motion planning calculates a valid path from a robot's start to goal state. This function is latency-critical and its performance limits the robustness and capabilities of robots. A performance gap of an order of magnitude has emerged in motion planning and control: robot joint actuators react at kHz rates, but promising online techniques for complex robots e.g., manipulators, quadrupeds, and humanoids (Figure 1), are limited to 100s of Hz by state-of-the-art software [7, 25].

Shrinking this performance gap will enable roboticists to explore longer planning horizons for robots, increasing their resilience to disturbances and unlocking new behaviors.

Hardware acceleration can address this challenge, but traditional hardware design can be tedious, iterative, and costly. It is essential to formalize design flows to keep development agile [11] as applications and robot platforms evolve.

2. Limitations of the State of the Art

Current robotics software libraries [15, 4, 21, 9, 19, 12] require at least an order of magnitude faster performance to enable emerging online motion planning and control techniques, like nonlinear model predictive control (MPC) [6, 14, 30, 23], to approach the kHz speeds at which robot actuators can respond [7, 25]. This gap persists despite their use of software templating and code generation to optimize functions for a particular robot model [4, 19]. For example, the gradient of rigid body dynamics [8, 10, 3], a key compute-bound kernel, takes up to 30% to 90% of the total runtime of promising nonlinear MPC systems [25, 24, 3, 23].

Relatively little work in hardware acceleration has been done for motion planning. Most robotics hardware accelerators have focused on other problems, such as perception and localization [5, 26, 29]. The few hardware solutions for motion planning are largely focused on the problem of collision-detection [20, 17]. They typically target systems with simple dynamics, e.g., cars and drones [27], and do not address the bottleneck of rigid body dynamics and its gradient.

For all hardware solutions the paramount challenge remains to make the design process efficient and flexible, providing *systematic methodologies for hardware development* that can generalize across different robots and algorithms.

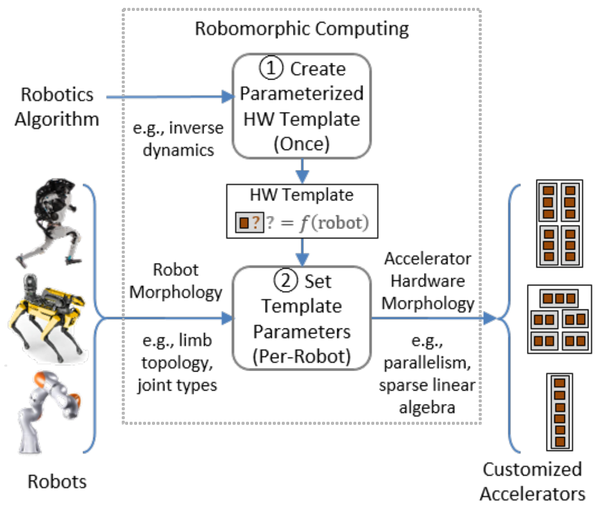

Figure 1: Overview of robomorphic computing, a design methodology to transform robot morphology into customized accelerator hardware morphology by exploiting robot features such as limb topology and joint type. This methodology can be applied to a wide variety of complex robots. Pictured are the Atlas [1], Spot [2], and LBR iiwa [16] robots, as examples.

3. Key Insights

Key insights in our paper are: (1) per-robot optimization techniques, which deliver state-of-the-art performance for robotics software [4, 19, 22], can be extended to hardware to deliver superior performance; and (2) these techniques can be formulated as a design methodology for domain-specific accelerators, to systematically customize accelerator hardware based on robot morphology: *robomorphic computing*.

Our design methodology (summarized in Figure 1) introduces a mapping between the physical structure of a robot and basic architectural primitives such as parallelism and data structure sparsity. In the robomorphic computing design flow: (1) a parameterized hardware template is created for a robotics algorithm *once*, exposing parallelism and matrix sparsity; then, (2) *for each robot*, template parameters are set according to the robot morphology, e.g., limb topology and joint types, creating an accelerator customized to that robot model.

This work provides a roadmap for future hardware accelerators for robotics. Our design flow provides a reliable pathway towards identifying useful algorithmic features in robotics applications, and a mechanistic way of encoding them in hardware. This relieves the burden of hardware designers in approaching new algorithms and robots.

4. Main Artifacts

We present: (i) a methodology to systematically design hardware accelerators customized to robot morphology; and what we believe is (ii) the first hardware accelerator for the rigid body dynamics gradient, designed with that methodology.

Robomorphic Computing Methodology. We detail our methodology and apply it to the design of our hardware accelerator, following the steps shown in Figure 1.

In Step 1, we expose parallelism in algorithm loops iterating over robot limbs and links, and map it to parallel processing elements in the hardware template. We identify linear algebra operations on key sparse robot matrices, e.g., joint transformations, and map those to functional units where constant values and the structure of operations on sparse data structures are parameterized by the robot links and joint types.

In Step 2, we use the numbers of limbs and links in the robot to instantiate parallel datapaths in the accelerator template. We use link inertia values and joint types to set constants and streamline operations in functional units, e.g., pruning multipliers and adders from sparse matrix-vector multiplications.

Accelerator for Robot Dynamics Gradient. We implement our dynamics gradient accelerator design on an FPGA for the iiwa manipulator (Figure 1), and integrate the accelerator in a coprocessor system connected to a host CPU, as it would be deployed for an off-the-shelf solution today. We evaluate the performance of our accelerator compared to state-of-the-art CPU and GPU baselines [4, 25] (see Figure 2).

We also synthesize an ASIC implementation using a 12 nm node, evaluating further benefits from a system on chip.

5. Key Results

Our FPGA accelerator achieves speedups of 8× and 86× over state-of-the-art CPU and GPU latency, and maintains an overall speedup of 1.9× to 2.9× when deployed in a coprocessor system (Figure 2). ASIC synthesis indicates an additional 7.2× factor over our FPGA implementation.

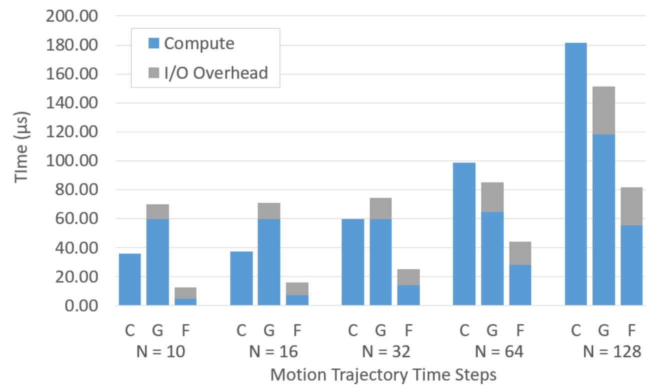

Figure 2: Our robot dynamics gradient accelerator on FPGA (F) achieves speedups of $2.2\times$ to $2.9\times$ over CPU (C) and $1.9\times$ to $5.5\times$ over GPU (G). Times are coprocessor system latency for a range of motion planning trajectory time horizons.

6. Contributions

The key contributions of this work include:

- Robomorphic computing: a new general methodology for the co-design of hardware accelerator architectures based on the high-level physical topology of a robot;
- Design of the first domain-specific accelerator for the gradient of rigid body dynamics; and
- Discussion of how our design methodology generalizes to more complex robot platforms, e.g., quadrupeds and humanoids, and other computational kernels in robotics.

Robomorphic computing provides a systematic and reliable shortcut to the traditional hardware accelerator design process, which is otherwise tedious, error-prone, and requires substantial intervention from domain experts.

Our accelerator for the gradient of rigid body dynamics represents meaningful progress towards real-time, online motion planning and control for complex robots, the performance of which is limited by current software solutions.

Using robomorphic computing to shrink this performance gap will allow robots to plan further into the future, helping them to safely interact with people in dynamic, unstructured, and unpredictable environments. This is a critical step towards enabling robots to realize their potential to address important societal challenges from elder care [13, 28], to the health and safety of humans in hazardous environments [18, 31].

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant 2030859, Grant DGE1745303, and Grant 1718160 and the Defense Advanced Research Projects Agency under Grant HR001118C0018. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations.

References

[1] Boston Dynamics. Atlas® | boston dynamics, Accessed in 2020. Available: bostondynamics.com/atlas.

[2] Boston Dynamics. Spot® | boston dynamics, Accessed in 2020. Available: bostondynamics.com/spot.

[3] Justin Carpentier and Nicolas Mansard. Analytical derivatives of rigid body dynamics algorithms. *Robotics: Science and Systems*, 2018.

[4] Justin Carpentier, Guilhem Saurel, Gabriele Buondonno, Joseph Mirabel, Florent Lamiraux, Olivier Stasse, and Nicolas Mansard. The pinocchio c++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In *2019 IEEE/SICE International Symposium on System Integration (SII)*, pages 614–619. IEEE, 2019.

[5] Yu-Hsin Chen, Joel Emer, and Vivienne Sze. Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks. In *ISCA*. ACM/IEEE, 2016.

[6] Moritz Diehl, Hans Joachim Ferreau, and Niels Haverbeke. Efficient numerical methods for nonlinear mpc and moving horizon estimation. In *Nonlinear model predictive control*, pages 391–417. Springer, 2009.

[7] Farbod Farshidian, Edo Jelavic, Asutosh Satapathy, Markus Giftthaler, and Jonas Buchli. Real-time motion planning of legged robots: A model predictive control approach. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 577–584. IEEE, 2017.

[8] Roy Featherstone. *Rigid body dynamics algorithms*. Springer, 2008.

[9] Martin L. Felis. RBDL: an efficient rigid-body dynamics library using recursive algorithms. *Autonomous Robots*, pages 1–17, 2016.

[10] Markus Giftthaler, Michael Neunert, Markus Stäuble, Marco Frigerio, Claudio Semini, and Jonas Buchli. Automatic differentiation of rigid body dynamics for optimal control and estimation. *Advanced Robotics*, 31(22):1225–1237, 2017.

[11] John L Hennessy and David A Patterson. A new golden age for computer architecture. *Communications of the ACM*, 2019.

[12] Michael G Hollars, Dan E Rosenthal, and Michael A Sherman. SD/FAST user's manual. *Symbolic Dynamics Inc*, 1991.

[13] Claudia Kalb. Could a robot care for grandma? *National Geographic*, Jan 2020.

[14] Jonas Koenemann, Andrea Del Prete, Yuval Tassa, Emanuel Todorov, Olivier Stasse, Maren Bennewitz, and Nicolas Mansard. Whole-body model-predictive control applied to the hrp-2 humanoid. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3346–3351. IEEE, 2015.

[15] Twan Koolen and Robin Deits. Julia for robotics: Simulation and real-time control in a high-level programming language. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 604–611. IEEE, 2019.

[16] KUKA AG. Lbr iiwa | kuka ag, Accessed in 2020. Available: kuka.com/products/robotics-systems/industrial-robots/lbr-iiwa.

[17] Shiqi Lian, Yinhe Han, Xiaoming Chen, Ying Wang, and Hang Xiao. Dadu-p: A scalable accelerator for robot motion planning in a dynamic environment. In *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2018.

[18] Courtney Linder. A cave is no place for humans, so darpa is sending in the robots. *Popular Mechanics*, Aug 2019.

[19] Frigerio Marco, Buchli Jonas, Darwin G Caldwell, and Semini Claudio. RobCoGen: a code generator for efficient kinematics and dynamics of articulated robots, based on domain specific languages. *Journal of Software Engineering in Robotics*, 7(1):36–54, 2016.

[20] Sean Murray, William Floyd-Jones, Ying Qi, George Konidaris, and Daniel J Sorin. The microarchitecture of a real-time robot motion planning accelerator. In *MICRO*. IEEE/ACM, 2016.

[21] Maximilien Naveau, Justin Carpentier, Sébastien Barthelemy, Olivier Stasse, and Philippe Souères. Metapod: Template meta-programming applied to dynamics: Cop-com trajectories filtering. In *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*, pages 401–406. IEEE, 2014.

[22] Sabrina M Neuman, Twan Koolen, Jules Drean, Jason E Miller, and Srinivas Devadas. Benchmarking and workload analysis of robot dynamics algorithms. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019.

[23] Michael Neunert, Cédric De Crousaz, Fadri Furrer, Mina Kamel, Farbod Farshidian, Roland Siegwart, and Jonas Buchli. Fast nonlinear model predictive control for unified trajectory optimization and tracking. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 1398–1404. IEEE, 2016.

[24] Zherong Pan, Bo Ren, and Dinesh Manocha. Gpu-based contact-aware trajectory optimization using a smooth force model. In *Proceedings of the 18th annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, page 4. ACM, 2019.

[25] Brian Plancher and Scott Kuindersma. A performance analysis of parallel differential dynamic programming on a gpu. In *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2018.

[26] Brandon Reagen, Paul Whatmough, Robert Adolf, Saketh Rama, Hyunkwang Lee, Sae Kyu Lee, José Miguel Hernández-Lobato, Gu-Yeon Wei, and David Brooks. Minerva: Enabling low-power, highly-accurate deep neural network accelerators. In *ISCA*. ACM/IEEE, 2016.

[27] Jacob Sacks, Divya Mahajan, Richard C Lawson, and Hadi Esmaeilzadeh. Robox: An end-to-end solution to accelerate autonomous control in robotics. In *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*, pages 479–490. IEEE, 2018.

[28] Jonathan Shaw. The coming eldercare tsunami. *Harvard Magazine*, Jan 2020.

[29] Amr Suleiman, Zhengdong Zhang, Luca Carlone, Sertac Karaman, and Vivienne Sze. Navion: a fully integrated energy-efficient visual-inertial odometry accelerator for auto. nav. of nano drones. In *VLSI Circuits*. IEEE, 2018.

[30] Yuval Tassa, Tom Erez, and Emanuel Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 4906–4913. IEEE, 2012.

[31] Guang-Zhong Yang, Bradley J. Nelson, Robin R. Murphy, Howie Choset, Henrik Christensen, Steven H. Collins, Paolo Dario, Ken Goldberg, Koji Ikuta, Neil Jacobstein, Danica Kragic, Russell H. Taylor, and Marcia McNutt. Combating covid-19—the role of robotics in managing public health and infectious diseases. *Science Robotics*, Mar 2020.