Thesis Proposal
for
Reducing Execution Time of Robot Motion Planning Algorithms with RISC-V ISA

Anthony JW Kenny

Electrical Engineering
Advisor: Vijay Janapa Reddi

March 1st, 2020
Version: 4.1

Abstract

This thesis aims to design RISC-V computer architecture that supports the fast execution of motion planning algorithms for drone applications. First, the computation of sampling-based motion planning algorithms commonly used in autonomous drones (such as Rapidly-exploring Random Tree (RRT), Rapidly-exploring Random Tree Star (RRT*), Probabalistic Road Map (PRM)) will be profiled on an unmodified RISC-V processor. From this profiling, common bottlenecks and hotspots in execution will be identified. Based on these results, this project will extend the RISC-V Instruction Set Architecture (ISA) and design a modified processor to support the extensions.

1 Project

This thesis aims to improve the speed of robotic motion planning algorithms at runtime by altering the architecture on which these algorithms run. My approach will explore the possibilities of extending the Instruction Set Architecture and altering the microarchitecture to achieve this speed-up. By working with the RISC-V ISA, which is open-source and extendable, I will be able to explore optimization through extending the ISA and the number of instructions.

1.1 Background and Motivation

Robotics now have the potential to shape the world in ways much more profound than ever seen before, through a wide range of applications. Central to robots being useful in these applications is the need for real-time motion planning, since they are no longer confined to static environments like an assembly line, and now need to adapt

to complex changing environments. Motion planning algorithms running on existing hardware is typically too slow currently for robots to be able to perform in these applications. There is a limitation on how efficiently these algorithms can be designed, or implemented in code. Thus, there exists the opportunity for significant speedups at the computer architecture level.

1.2 Specific Problem and Metrics of Success

Modern robotics applications require run-time motion planning, which is implemented with computationally complex algorithms. There is a limit to how efficiently an algorithm can be designed to solve these problems, and so, how can I work in the computer architecture space to increase the speed with which these algorithms run on hardware.

While the specific metrics of success will be dependent on the algorithm I eventually select, I will measure and verify based on the following aspects:

- **Extended Instruction Set Architecture** (if applicable) is well defined and correct
- **Modified processor design** (if applicable) produces correct output for a given input, and will reliably compute correct paths.
- **Faster CPU performance** for the implementation of the selected algorithm. This can be measured through the CPU performance equation:
  - Instructions/program is an area that could be targeted through extending the ISA
  - Cycles/instruction could be brought down through a modified processor design
  - Seconds/cycle is probably something that will remain constant and is beyond the scope of my intended optimizations.

  While it is hard to define a specific speed-up benchmark, my research indicates that different methods of using hardware to quicken motion planning algorithms yield speed-ups from several times to several orders of magnitude.
- **Power use** is such that total energy consumption declines by nature of the increase in speed of execution.
- **Area of chip** is not prohibitive for potential applications.
- **Latency** is lower than benchmark tests (again, specific amount of speedup will depend on the algorithm that is eventually selected).

2 End User

The use of robotics across different industries is increasing rapidly, with global sales of industrial robotics doubling over the last 5 years (World Robotics Report, 2018). There are many factors driving this increase, such as the concept of "Co-Bots" (collaborative autonomous robots), Robotics as a Service (RaaS) businesses, the integration of AI into robotics, and the rise of the drones in many varied applications.
The overall problem that end users of robotics face is that current motion planning

algorithms cannot execute at run time fast enough to adapt to changing working environments. The significance of working in computer architecture to speed up these algorithms is such that it will extend the situations in which robots can be useful.
The specific end user of this application may vary based on the final algorithm chosen, but more generally the system developed for this thesis could find use in a wide variety of final robotics applications. More specifically, the concern for speed in this thesis is particularly focussed on run-time computation. Thus, within the space of robot motion planning, it is most likely to see application where require robots to adapt to their changing environments.
At any rate, the extended ISA and/or adapted microarchitecture will sit within larger robotic systems to provide the specific functionality of efficiently computing motion planning.

3 Existing Solutions and Previous Work

3.1 Previous Work

- *Robot Motion Planning on a Chip:* constructs robot-specific circuitry for motion planning that calculates plans approximately 3 orders of magnitude faster than existing methods. It uses parallelized collision detection for each edge of a robotic arm, so that the time taken to determine potential collisions with an edge is independent of the number of edges. Prototyped in Verilog HDL and compiled circuit design onto an FPGA for testing. (Murray, S., Floyd-Jones, W., Qi, Y., Sorin, D., Konidaris, G., 2016)
- *Robot Motion Planning on a Chip:* constructs robot-specific circuitry for motion planning that calculates plans approximately 3 orders of magnitude faster than existing methods. It uses parallelized collision detection for each edge of a robotic arm, so that the time taken to determine potential collisions with an edge is independent of the number of edges. Prototyped in Verilog HDL and compiled circuit design onto an FPGA for testing. (Murray, S., Floyd-Jones, W., Qi, Y., Sorin, D., Konidaris, G., 2016)
- *High-throughput Computation of Shannon Mutual Information on Chip:* uses multiple cores to compute Shannon mutual information without the use of approximation. When implemented on an FPGA, this architecture computes approximately 2 orders of magnitude faster and consumes an order of magnitude less power. (Zhi Xuan Li, P., Zhang, Z., Karaman, S., Sze, V.)
- *FPGA based Combinatorial Architecture for Parallelizing RRT:* aims to accomplish more per clock cycle in the application of RRT. They use FPGA combinatorial architecture that allows multiple RRTs to work together to explore a map. (Malik, G. S., Gupta, K., Krishna, K. M., Chowdhury, S. R. 2015)
- *A Motion Planning Processor on Reconfigurable Hardware:* studies the possibility of an FPGA based motion planning processor and evaluates its performance in executing the feasibility checks associated with motion planning algorithms. (Atay, N., Bayazit, B. 2006)

- *A Programmable Architecture for Robot Motion Planning Acceleration:* developed a microarchitecture of an accelerator for collision detection and graph search, implemented in Verilog and using an FPGA as a platform.. (Murray, S., Floyd-Jones, W., Konidaris, G., Sorin, D.)
- *MAVBench: Micro Aerial Vehicle Benchmarking:* introduces a framework to foster the development of autonomous Micro Aerial Vehicles, consisting of a simulator and application benchmark suite. It shows that efficient system design can improve MAV battery life by up to 1.8x. (Boroujerdian, B., Genc, H., Krishnan, S., Cui, W., Faust, A., Reddi, V. J. 2019)
- *Towards Deep Learning using TensorFlow Lite on RISC-V:* presented the software infrastructure for optimizing the execution of neural network calculations by extending the RISC-V ISA. A small number of instruction extensions achieved coverage over a wide variety of speech and vision application deep neural networks. (Louis, M. S., Azad, Z., Delshadtehrani, L., Gupta, S., Warden, P., Reddi, V. J., Joshi, A. 2019)
- *GAP-8: A RISC-V SoC for AI at the Edge of the IoT:* proposed a programmable RISC-V computing engine with 8-core and convolutional neural network accelerator for power efficient, battery operated, IoT edge-device computing. (Flamand, E., Rossi, et al. 2018)

3.2 Existing Technology

FPGA: My research shows that previous work in speeding up motion planning algorithms has made extensive use of Field Programmable Gate Arrays as a platform, which allows for easy prototyping of microarchitecture designs.

RISC-V: Is an open-source, extendable Instruction Set Architecture based on the reduced instruction set computer principles. By being extendable, it allows for potential optimizations within the ISA, as shown in the final two papers listed above in my research.

4 New, Innovative, Interesting Approach

While optimizing hardware for motion planning is a very popular area of research, from my research nobody has attempted to do so by extending the RISC-V Instruction Set Architecture.

Looking at the field of Computer Architecture, I anticipate making changes to the ISA or the processor in order to improve the performance of a specific algorithm.

Considering the CPU Performance Equation;

$$\text{CPU Execution Time} = \frac{\text{instructions}}{\text{program}} \times \frac{\text{cycles}}{\text{instruction}} \times \frac{\text{seconds}}{\text{cycle}} = \frac{\text{seconds}}{\text{program}}$$

One potential approach to improve the CPU speed of a motion planning algorithm is to reduce instructions per program, through extending the RISC-V ISA. This is an approach that I've found has been used to optimise Deep Learning and AI on edge-computing IoT devices, but not in the robot motion planning area.

Another avenue through which the CPU speed could be increased is be reducing Cycles/Instruction. This would be done by altering the microarchitecture itself, with the aim of lowering how many cycles are required to execute instructions commonly used in motion planning algorithms.

5 Related Courses and Required Skills

5.1 Courses

Course	**Relevant Concepts & Experience**
CS141: Computing Hardware	Digital Logic, Basic Computer Architecture, Machine Code, ISAs, Processor Design, FPGA, HDLs.
CS146: Computer Architecture	Processor Design, Computer Organization, quantitative evaluation of design alternatives.
CS182: Artificial Intelligence	Background understanding of motion planning algorithms and their complexity.
ES153: Laboratory Electronics	Basics of processor and computer design, assembly and machine language.
ES96r: Engineering Problem Solving	Project management, defining problem space,techniques to find solutions to a problem.
ES150: Probability for Engineering	Fundamental understanding of probability, which is often used in robot motion planning algorithms.
CS50 & CS51: Introduction to CS	Basic programming skills, version control.
CS61: Systems Programming and Machine Organization	Fundamentals of low-level machine organisation, C and machine language.
Physics 12 A & B	Python and matlab for data collection, manipulation, visualization and analysis, which will assist with the Measure & Verify stage of this thesis.

5.2 New Skills

- Familiarity with the RISC-V Instruction Set Architecture and extending it
- Aladdin: a simulator tool that will help with research
- Firesim: another simulation framework
- Vivado Design Suite from Xlinix

6 Preliminary Project Milestones

Project Milestone	Completion Date
Project Proposal Due	9 September 2019
Set up testing and development environment	18 September 2019
Determine most suitable algorithm to optimize	31 September 2019
Checkpoint 1: Design Specs	4 October 2019
Categorize algorithm's performance in hardware and define bottlenecks/key areas suitable for optimization	10 October 2019
Design initial solution	25 October 2019
Checkpoint 2: Design Documentation	1 November 2019
Complete initial solution implementation	20 November
Measure and verify comparative performance	1 December 2019
Checkpoint 3: Mid Year Report	4 December 2019
Design and complete implementation 2	25 January 2020
Checkpoint 4: Poster Session and Peer Reviews	31 January 2020
Measure and verify comparative performance	15 February 2020
Checkpoint 5: Report Draft	28 February 2020
Final Oral Presentation	24 March 2020
Final Written Report	3 April 2020
Final Poster	6 May

7 Budget

Item	Link	Cost
Zynq Board	https://www.xilinx.com/products/boards-and-kits/1-elhabt.html	$495

8 List of Acronyms

ISA Instruction Set Architecture

PRM Probabalistic Road Map

RRT Rapidly-exploring Random Tree

RRT* Rapidly-exploring Random Tree Star