Carnegie Mellon University

FPGA-AMP

Design Review

Matt Ngaw, Yufei Shi, Chris Stange
Team C5

Background

Motion planning

- Critical step in the robotics pipeline
- Guides motion of the robot
- Rapidly-exploring Random Tree (RRT) finds collision-free trajectories from a start position to a goal position
- Run A* or Dijkstra's on the set of collision-free trajectories in order to find optimal route

Problem

- For complex, latency sensitive robots, RRT is too slow on CPUs and too power-inefficient on GPUs

Solution

- Use FPGAs to accelerate motion planning while also consuming less power

https://en.m.wikipedia.org/wiki/File:Rapidly-exploring_Random_Tree_%28RRT%29_500x373.gif

Carnegie Mellon University

Quantitative Design Requirements

Relative to the reference implementation on AMD Ryzen 9 5950X

95% accuracy

- Similar but not exact same motion plan (RRT is a non-deterministic algorithm)

10x speedup (latency)

- Time elapsed while servicing motion planning query
- Prior academic research achieved 1000x speedup
  - However, the comparison is questionable
- 10x is a modest speedup that justifies the addition of an FPGA

70% less power

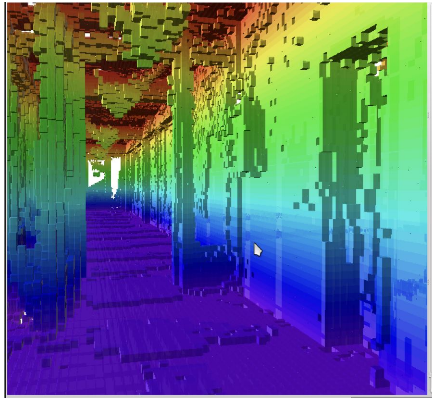
- 5950X consumes 105 W, FPGA use 30-40W

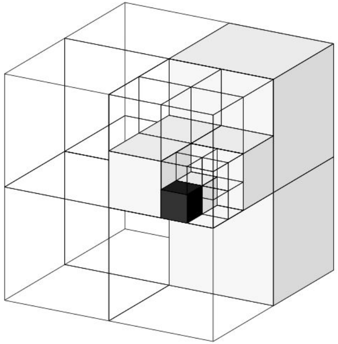
98% less energy

- Assuming the speedup and power requirements hold

Carnegie Mellon University

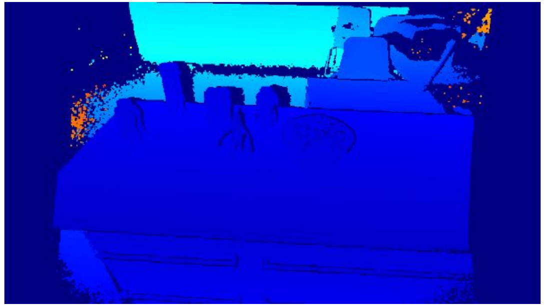
Solution Approach & System Specification

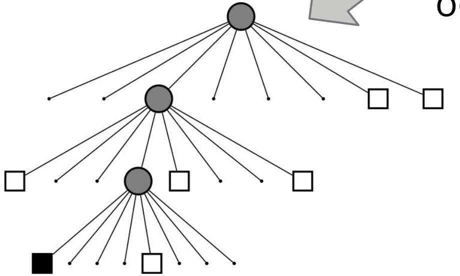




voxels

libfreenect2 +
iai_kinect2





octree

octomap

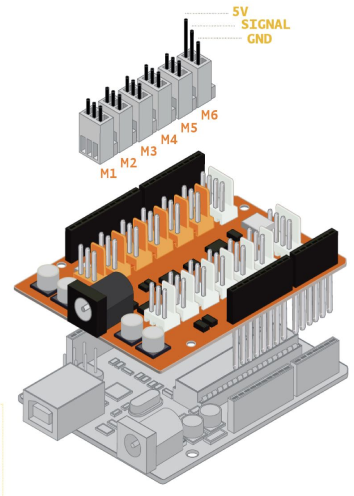
https://github.com/code-iai/iai_kinect2

https://octomap.github.io/

http://www.arminhornung.de/Research/pub/hornung13auro.pdf

Carnegie Mellon University

Solution Approach & System Specification



http://www.arminhornung.de/Research/pub/hornung13auro.pdf

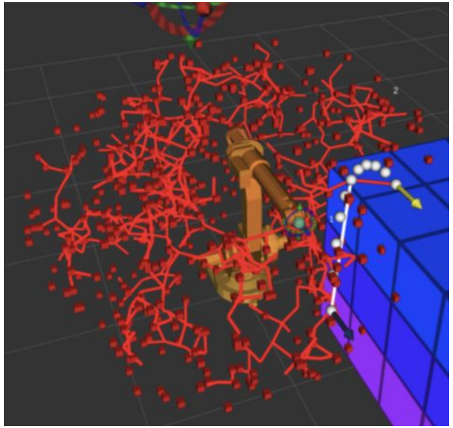
http://msl.cs.uiuc.edu/~lavalle/papers/LavKuf01.pdf

https://arxiv.org/pdf/1911.04676.pdf

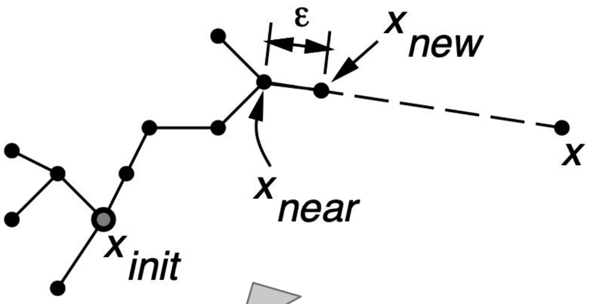
https://cdn.robotshop.com/media/A/Ard/RB-Ard-81/pdf/arduino-braccio-robotic-arm-quick-start-guide.pdf

dynamics

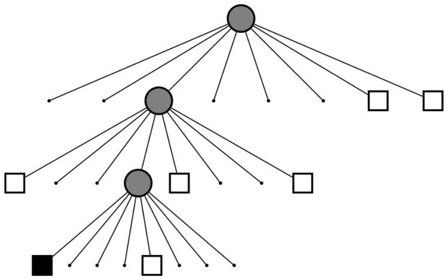
MOTOR "1" BASE
MOTOR "2" SHOULDER
MOTOR "3" ELBOW
MOTOR "4" VERTICAL WRIST
MOTOR "5" ROTATORY WRIST
MOTOR "6" GRIPPER



motion plan (kinematics)



RRT (*)



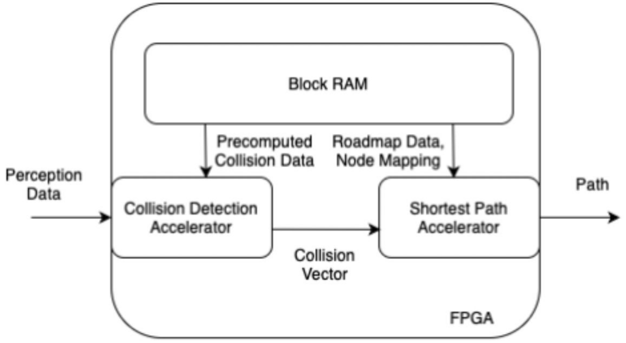
Block Diagram

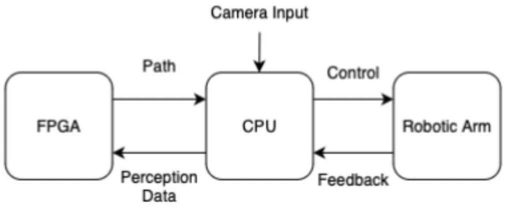
FPGA-AMP Accelerator (Top Diagram)

- Receives collision data in the form of voxels from the perception module on the CPU
- Accelerates RRT and uses A* to find the shortest path
- Returns generated path to inverse-kinematics module on CPU
- Will be tested in software simulation as well as integrated and tested in the full system

Full System (Bottom Diagram)

- Used to test FPGA-AMP accelerator in real world scenarios with a real robotic arm and perception data
- Cameras and robotic arm communicate with and are controlled by CPU
- CPU runs ROS (Robot Operating System)
- FPGA is used as offload accelerator for motion planning tasks.





Carnegie Mellon University

Implementation Plan

Pick and place

- Field of obstacles
- Targets will be placed in hard to reach spots
- Simulate real world environments (i.e car factory)
- Dynamic vs. Static scenarios

Perception

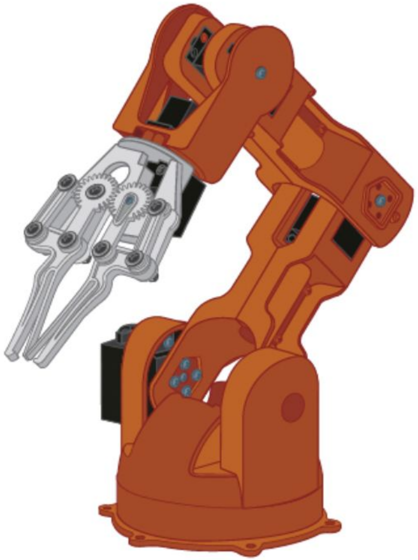
- Open source library using Xbox One Kinect depth sensor

FPGA

- Design and implement our own motion planning accelerator
- Xilinx FPGA-CPU. Packaged together for fast compute offloading. Fast I/O makes the offload feasible.
- Vitis HLS for fast iteration times

Robotic Arm Control

- Open source library for Inverse-Kinematics



https://cdn.robotshop.com/media/A/Ard/RB-Ard-81/pdf/arduino-braccio-robotic-arm-quick-start-guide.pdf

Carnegie Mellon University

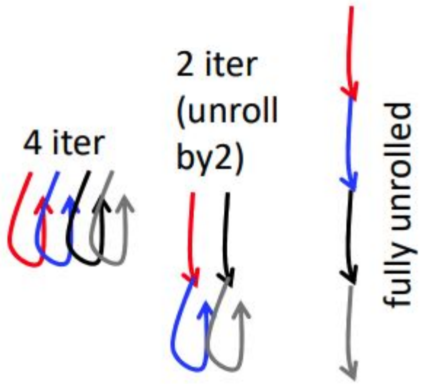
Hardware Kernel Optimizations

Loop Unrolling

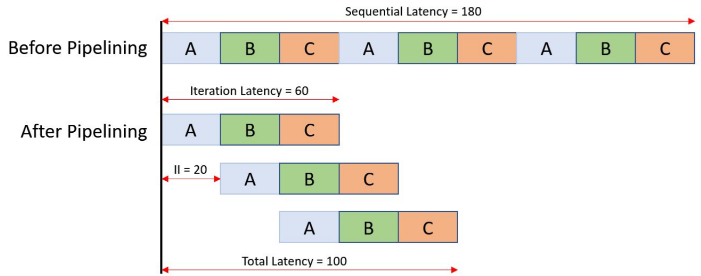
- 99% of instructions executed during RRT are collision detection
  - Collision detection is massively parallel
- Unrolls the loop so that each loop iteration is done in parallel

Pipelining

- Overlap execution
- Break up critical path and improve utilization
- Higher clock frequency



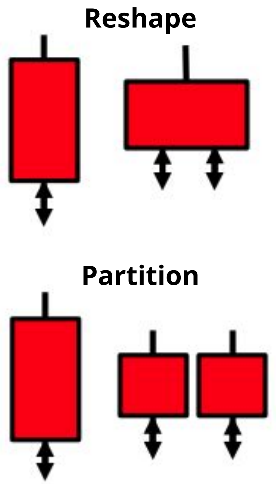
Professor James Hoe, 18-643 F23, Lecture 8
https://users.ece.cmu.edu/~jhoe/course/ece643/latest/L08.pdf



Xilinx Docs, Vitis HLS User Guide, Pipelining Paradigm
https://docs.xilinx.com/r/en-US/ug1399-vitis-hls/Pipelining-Paradigm

Carnegie Mellon University

Memory & I/O Optimizations



Professor James Hoe, 18-643 F23, Lecture 8
https://users.ece.cmu.edu/~jhoe/course/ece643/latest/L08.pdf

Buffering On-Chip

- Limit number of trips to DRAM
- Make full use of DRAM bandwidth
- Buffer data that will soon be used
- e.g. Tiled matrix multiplication

Array Reshape

- Vary width of memory port

Array Partition

- Split up data across multiple memories for parallel access

Carnegie Mellon University

Testing, Verification, and Metrics

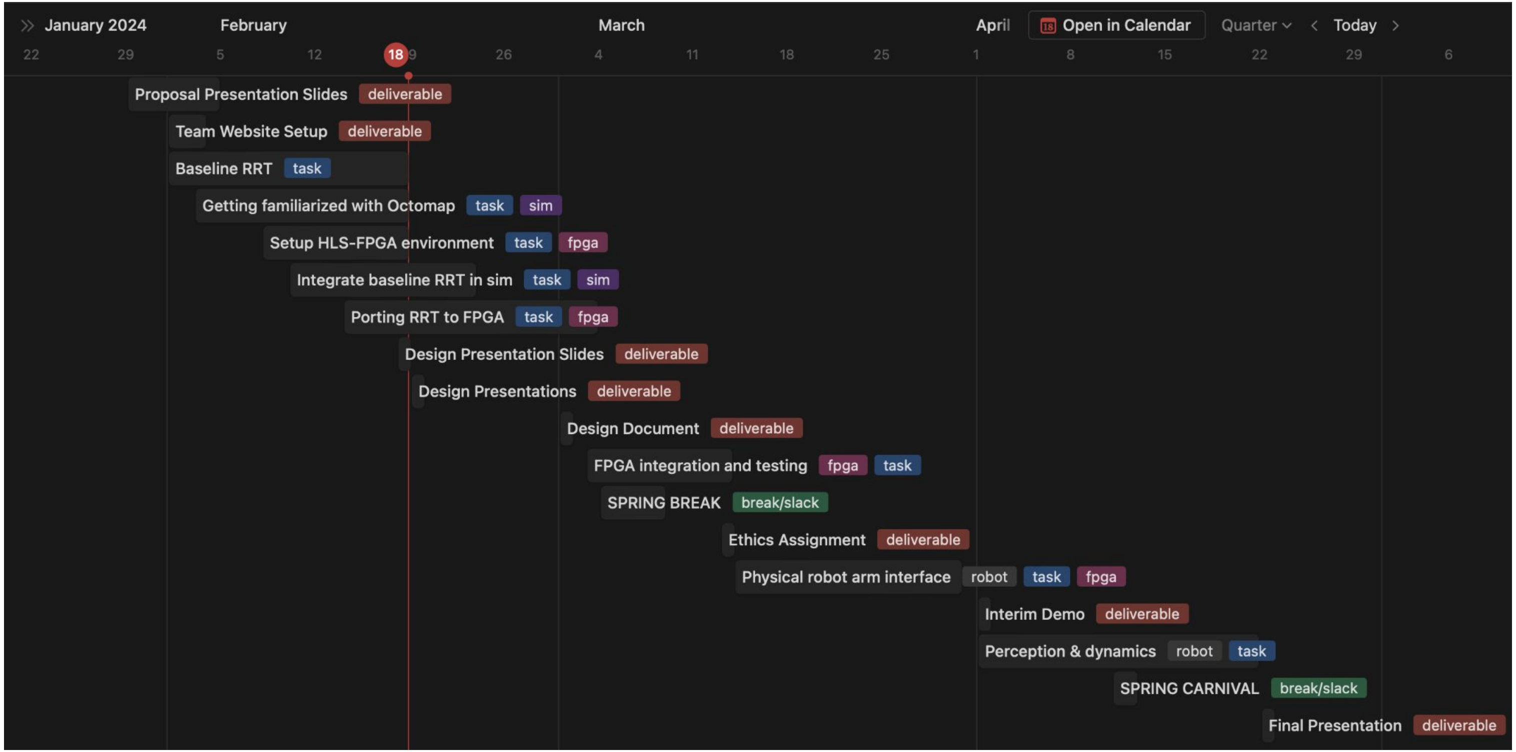
Requirements	Testing	Metrics
Generate Collision Free Paths	Check that the path generated avoids collisions	Avoid collisions on >95% of scenes tested
Generate Optimal Paths	Ensure the delta between the optimal path and the reference solution is similar to the delta between the optimal path and FPGA-AMP	Similar deltas on >95% of scenes tested
Low Latency	FPGA-AMP generates paths significantly faster than the reference solution	10x speedup vs. reference solution
Power Efficient	FPGA-AMP generates paths while being significantly more power efficient than reference solution	70% decrease in power consumption vs. reference solution
Energy Efficient	FPGA-AMP generates paths while being significantly more energy efficient than reference solution	98% decrease in energy consumption per path vs. reference solution

Note: All requirements will be tested in both simulation and with real hardware.

Tasks and Division of Labor

- Baseline RRT (Yufei, Chris)
- **Simulation Environment (Yufei, Chris)**
- HLS-FPGA environment (Matt)
- **Uarch design (Matt, Yufei, Chris)**
- Porting RRT to **HLS (Matt)**
- Optimization (Matt, Yufei, Chris)
- **Perception (Yufei)**
- Robotic Arm **Dynamics (Chris)**
- Robotic Arm & FPGA Integration (Matt)
- Full System **Integration (Matt, Yufei, Chris)**

Schedule



Carnegie Mellon University