MULTI-VEHICLE MAP FUSION USING GNU RADIO OPTIMIZATION AND ACCELERATION OPPORTUNITIES

Augusto Vega
Akin Sisbot
Alper Buyuktosunoglu
Arun Paidimarri
David Trilla
John-David Wellman
Pradip Bose

IBM T. J. Watson Research Center
Acknowledgment

- Thanks to the many IBM colleagues who contribute to and support different aspects of this work + our esteemed university collaborators at Harvard, Columbia, and UIUC (Profs. David Brooks, Vijay Janapa Reddi, Gu-Yeon Wei, Luca Carloni, Ken Shepard, Sarita Adve, Vikram Adve, Sasa Misailovic) + many brilliant graduate students and postdocs!

- Special thanks to Dr. Thomas Rondeau, Program Manager of the DARPA MTO DSSoC Program

This research was developed, in part, with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. This document is approved for public release: distribution unlimited.
Outline

- **Part 1: DARPA-funded EPOCHS project**
  - Domain-specific (heterogeneous) SoC development

- **Part 2: EPOCHS Reference Application (“ERA”)**
  - Application domain: *multi-vehicle cooperative perception*

- **Part 3: 802.11p Transceiver**
  - Optimization and acceleration opportunities
Outline

- **Part 1: DARPA-funded EPOCHS project**
  - Domain-specific (heterogeneous) SoC development

- **Part 2: EPOCHS Reference Application (“ERA”)**
  - Application domain: *multi-vehicle cooperative perception*

- **Part 3: 802.11p Transceiver**
  - Optimization and acceleration opportunities
DARPA’s Domain-Specific System on Chip (DSSoC) Program
Program Manager: Dr. Tom Rondeau

- **Goal:** to develop a heterogeneous system-on-chip (SoC) comprised of many cores that mix general purpose processors, special purpose processors, hardware accelerators, memory, and input/output (I/O) devices to significantly improve performance of applications within a **domain**.

- **A domain is larger than any one application**
  - We target the “super” domain of embedded processors for **autonomous/connected cars**

*Source: https://www.darpa.mil/program/domain-specific-system-on-chip*
Application Domain: Cooperative Perception

- Automakers use arrays of sensors to build redundancy into their systems

This Image is Why Self-Driving Cars Come Loaded with Many Types of Sensors

When’s a pedestrian not a pedestrian? When it’s a decal.

Source: MIT Technology Review
Automakers use arrays of sensors to build redundancy into their systems.

We propose a complementary approach: **multi-vehicle (cooperative) perception**
- Cars exchange locally-generated maps
- Each vehicle merges its local map and the received ones in real time.

This Image is Why Self-Driving Cars Come Loaded with Many Types of Sensors

When’s a pedestrian not a pedestrian? When it’s a decal.

Source: MIT Technology Review
Efficient Programmability Of Cognitive Heterogeneous Systems

“EPOCHS” → our proposed solution for the design challenge presented by the DSSoC program
Efficient Programmability Of Cognitive Heterogeneous Systems

“EPOCHS” → our proposed solution for the design challenge presented by the DSSoC program

Agile methodology to quickly design and implement an easily programmed domain-specific SoC for real-time cognitive decision engines in connected vehicles

“Super”-Domain: Software-Defined Radio + Computer Vision

10X – 100X reduction in person-years

FPGA prototype, emulation, optimization, software bring-up
Efficient Programmability Of Cognitive Heterogeneous Systems

“EPOCHS” → our proposed solution for the design challenge presented by the DSSoC program

Agile methodology to quickly design and implement an easily programmed domain-specific SoC for real-time cognitive decision engines in connected vehicles

“Super”-Domain: Software-Defined Radio + Computer Vision

10X – 100X reduction in person-years

FPGA prototype, emulation, optimization, software bring-up
The Big Picture (Where Does This Talk Fit In?)

DSSoC’s Full-Stack Integration

- Development Environment and Programming Languages
- Application
- Libraries
- Operating System
  - Heterogeneous architecture composed of Processor Elements:
    - CPUs
    - Graphics processing units
    - Tensor product units
    - Neuromorphic units
    - Accelerators (e.g., FFT)
    - DSPs
    - Programmable logic
    - Math accelerators

Multi-vehicle map fusion using GNU Radio
Outline

- **Part 1: DARPA-funded EPOCHS project**
  - Domain-specific (heterogeneous) SoC development

- **Part 2: EPOCHS Reference Application (“ERA”)**
  - Application domain: *multi-vehicle cooperative perception*

- **Part 3: 802.11p Transceiver**
  - Optimization and acceleration opportunities
ERA: EPOCHS Reference Application

- Multimodal sensing
- Local occupancy map generation
- DSRC-based V2V communication
- Real-time map fusion

Contribute!
https://github.com/IBM/era
ERA Main Components (Single Robot’s Viewpoint)

- Raw sensor data generated (simulated) using Gazebo in ERA v2
  - ERA v3 will replace Gazebo with an automotive simulation platform
ERA Main Components (Single Robot’s Viewpoint)

- Raw sensor data is first converted into laser scans which are used to generate a 2D occupancy grid map.
ERA Main Components (Single Robot’s Viewpoint)

- Occupancy grid maps are serialized, compressed and put into a GNU Radio PDU
- Outbound PDUs are injected into the 802.11p transceiver

Open-source implementation by Bastian Bloessl
https://github.com/bastibl/gr-ieee802-11
Locally- and remotely-generated occupancy maps are merged in real time to improve the accuracy of the surroundings’ view.

In ERAv2, *merging* is merely adding maps—Executed several times per second (!)

```cpp
GridPtr combineGrids (const vector<nm::OccupancyGrid>& grids, const double resolution)
{
    GridPtr combined_grid(new nm::OccupancyGrid());
    combined_grid->info = getCombinedGridInfo(grids, resolution);
    combined_grid->data.resize(combined_grid->info.width*combined_grid->info.height);
    fill(combined_grid->data.begin(), combined_grid->data.end(), -1);
    ROS_DEBUG_NAMED("combine_grids", "Combining %zu grids", grids.size());

    BOOST_FOREACH (const nm::OccupancyGrid& grid, grids) {
        for (coord_t x=0; x<grid.info.width; x++) {
            for (coord_t y=0; y<grid.info.height; y++) {
                const Cell cell(x, y);
                const signed char value=grid.data[cellIndex(grid.info, cell)];

                // Only proceed if the value is not unknown
                if ((value>=0) && (value<=100)) {
                    BOOST_FOREACH (const Cell& intersecting_cell,
                        intersectingCells(combined_grid->info, grid.info, cell)) {
                        const index_t ind = cellIndex(combined_grid->info, intersecting_cell);
                        combined_grid->data[ind] = max(combined_grid->data[ind], value);
                    }
                }
            }
        }
    }

    ROS_DEBUG_NAMED("combine_grids", "Done combining grids");
    return combined_grid;
}
```
Option 1: Two-Computer Setup

- One Gazebo instance simulating one single robot/vehicle in each computer
- Over-the-air 802.11p communication (10-MHz OFDM with up to 64-QAM modulation)
Option 2: Standalone Setup

- Runs on a single computer, replacing over-the-air communication with network sockets
- Easiest setup to start with ✓
Outline

- **Part 1: DARPA-funded EPOCHS project**
  - Domain-specific (heterogeneous) SoC development

- **Part 2: EPOCHS Reference Application (“ERA”)**
  - Application domain: *multi-vehicle cooperative perception*

- **Part 3: 802.11p Transceiver**
  - Optimization and acceleration opportunities
802.11p Transceiver within ERA

Open-source IEEE 802.11p transceiver implemented in GNU Radio by Bastian Bloessl

https://github.com/bastibl/gr-ieee802-11
802.11p Transceiver within ERA

**Open-source IEEE 802.11p transceiver implemented in GNU Radio by Bastian Bloessl**

https://github.com/bastibl/gr-ieee802-11

---

**TRANSMITTER FLOWGRAPH**

From ROS
- WiFi MAC
- WiFi Encoding and Packet Generation
- OFDM Carrier Allocation
- IFFT
- Cyclic Prefix Generation
  - To TX DAC

**RECEIVER FLOWGRAPH**

To ROS
- Packet Decoder and Mac
- OFDM Frame Equalizer
- FFT
- Sync Long
- Sync Short
  - From RX ADC

---

**Execution Time (%)**

- Functions identified for acceleration

---

IBM Research

February '20
Acceleration Options (for \textit{cexpf}): Preliminary Results

![Chart showing execution time per cexpf operation](chart)

- **Baseline**
- **FPGA with DMA**
- **CPU Vectorized**
- **Fully Optimized FPGA with DMA**

**Execution time per cexpf operation**

- Computation Time
- Mem Copy Overhead

**ARM Cortex-A53**
Acceleration Options (for cexpf): Preliminary Results

Execution time per cexpf operation

- Computation Time
- Mem Copy Overhead

CPU Cycles

CPU Baseline   FPGA w/DMA

ARM Cortex-A53  Xilinx UltraScaleMP+ ZYNQ ZCU102

FPGA Implementation
(dual data-path pipeline)

$(\cos b + \sin b i)$

e$^a + bi = e^a \times (\cos b + \sin b i)$
Acceleration Options (for cexpf): Preliminary Results

Execution time per cexpf operation

- Computation Time
- Mem Copy Overhead

CPU Cycles

<table>
<thead>
<tr>
<th></th>
<th>CPU Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Baseline</td>
<td>40</td>
</tr>
<tr>
<td>FPGA w/DMA</td>
<td>20</td>
</tr>
<tr>
<td>CPU Vectorized</td>
<td>15</td>
</tr>
</tbody>
</table>

FPGA Implementation (dual data-path pipeline)

\[ e^a + bi = e^a \ast (\cos b + \sin b \, i) \]
Acceleration Options (for \textit{cexpf}): Preliminary Results

**Execution time per cexpf operation**

- **Computation Time**
- **Mem Copy Overhead**

<table>
<thead>
<tr>
<th></th>
<th>CPU Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Baseline</td>
<td>35</td>
</tr>
<tr>
<td>FPGA w/DMA</td>
<td>25</td>
</tr>
<tr>
<td>CPU Vectorized</td>
<td>20</td>
</tr>
<tr>
<td>FPGA w/DMA (Fully Optimized)</td>
<td>15</td>
</tr>
</tbody>
</table>

**FPGA Implementation**

(dual data-path pipeline)

\[
e^{a+bi} = e^a \times (\cos b + \sin b i)
\]

- Fully-optimized implementation (idealized bound)
  - 300 MHz (instead of 100 MHz)
  - Four parallel computation engines
  - Memory-copy elimination

**ARM Cortex-A53**

**Xilinx UltraScaleMP+ ZYNQ ZCU102**

**ARM’s SIMD extension (NEON)**

- Floating-Point Unit: Float-to-Fixed Point
- CORDIC Unit: Sine & Cosine
- Floating-Point Unit: Fixed-to-Float Point
- Floating-Point Unit: Exponent
- Floating-Point Unit: Multiplier
ERA Roadmap

Version 1

Version 2

Version 3
ERA Roadmap

ERA is only intended to enable **cooperative automotive**, with support for DSRC, and 5G (future)

This makes ERA unique

---

**Layer 1: World Simulators** (sensor data source)
- CarSim
- Gazebo
- CARLA
- LGSV

**Layer 2: Automotive Platforms** (perception, plan and control)
- Apollo
- Autoware

**Layer 3: Cooperative Vehicles Platform** (swarming and V2X support)
- Apollo API
- Autoware API
- Multi-Vehicle Cooperation Logic
- 802.11p
- 5G

Some raw sensor data is directly fed to ERA
Summary

- The **domain-specific (heterogeneous) SoCs era** is here!

- DARPA’s Domain-Specific System on Chip (DSSoC) Program
  - Our proposed application domain: **multi-vehicle cooperative perception**
  - Local sensing + V2V communications
  - The **DSRC transceiver plays a critical role** for real-time V2V communications

**Performance**

**Power efficiency**

**ROS and GNU Radio “worlds” coexisting**
Summary

- The domain-specific (heterogeneous) SoCs era is here!

- DARPA’s Domain-Specific System on Chip (DSSoC) Program
  - Our proposed application domain: multi-vehicle cooperative perception
  - Local sensing + V2V communications
  - The DSRC transceiver plays a critical role for real-time V2V communications

- ROS and GNU Radio “worlds” coexisting

- Turn ERA into a benchmark for cooperative mobility that can be easily “plugged” into existing platforms

- Do you want to collaborate?
  - Contact: ajvega@us.ibm.com
  - GitHub: https://github.com/IBM/era