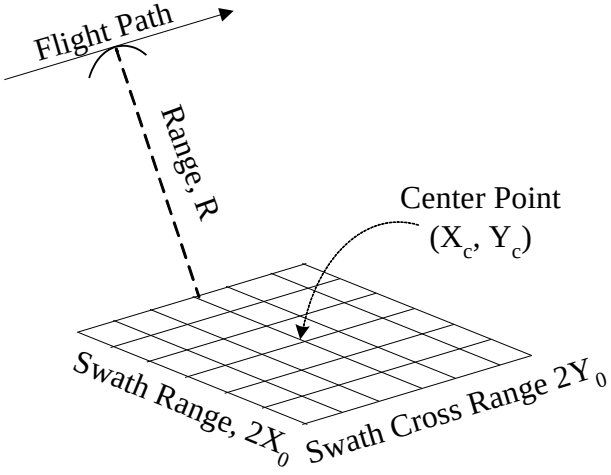
SAR: Synthetic Aperture Radar

Functions: FFT, image/matrix processing, knowledge formation, storage subsystem
Inputs: synthetic radar returns (statically generated)
Metrics: object identification accuracy, system bandwidth (per phase and per kernel)

1. Overview

Synthetic Aperture Radar (SAR) is a form of radar in which (typically speaking) a moving aerial platform scans a region with radio pulses perpendicular to its direction of travel. The system then composes an image of the region as an aggregate of the radar returns received as the platform travels along its flight path, which improves the image quality by integrating radar sub-images taken at multiple different angles. For further reading on this radar mode, please refer to Appendix A (page 46) of `HPCS-SSCA3-Knowledge-Formation.pdf`. Additional documentation is available at http://www.sarbenchmark.com/?page_id=17.



This benchmark implements a SAR processing system using an adaptation of a past project from MIT/Lincoln Labs; complete documentation of the internal workings of the system is available

in the `HPCS-SSCA3-Knowledge-Formation.pdf` document from that project. The benchmark is centered on four kernels, divided into two phases:

Phase 1: front-end (a) image formation and (b) storage (receiving and storing radar returns for later analysis);

Phase 2: back-end (c) image retrieval and (d) knowledge formation (locating and identifying objects of interest in the composite radar images).

2. System requirements

Platform: Ubuntu 18.04 LTS with gcc 7.4.0. Code may build and run successfully on other versions/platforms, but has not been tested with them.
Storage: ~300MB
Dependencies: none

3. Build and run

- Download and extract the zipfile from www.adacenter.org/milspec
- From the `SAR/` directory, `make clean && make test`
- Results are displayed in the terminal, as below:
    - The first line shows the SAR accuracy from kernel 4 (object detection/identification)
    - The performance results report time in seconds and bandwidth in MB/s (note that on a CPU-bound system, as below, the benchmark may take 30+ minutes to complete)

```
all> Overall 3641 compute test(s) passed, and 97 test(s) failed, out of a total of 3738 tests.
all> END STAGE 2 - KNOWLEDGE FORMATION.
all> HPCS SSCA #3 Sensor Processing and Knowledge Formation : END OF TEST


****PERFORMANCE RESULTS****

    kernel1_read Time:       10.389 s, Size:   314.2500 MB, BW:    30.2474 MB/s
       formimage Time:     1205.191 s, Size:   714.5869 MB, BW:     0.5929 MB/s
   kernel2_write Time:        9.835 s, Size:   211.1492 MB, BW:    21.4695 MB/s
     kernel3Read Time:       11.634 s, Size:   614.2039 MB, BW:    52.7942 MB/s
  kernel4Compute Time:      945.018 s, Size:   672.7102 MB, BW:     0.7118 MB/s
    kernel4Write Time:        5.680 s, Size:    58.4082 MB, BW:    10.2835 MB/s
    SSCA#3_total Time:     2188.480 s, Size:     0.0000 MB, BW:     0.0000 MB/s
```

Note that the provided SAR implementation will display some warnings or errors about (1) the scalable data generator being disabled, and (2) incorrect detection of objects in the simulated radar returns. These are expected; (1) because we pre-generate input data rather than generating new input data for each benchmark run (thus ensuring that test results are deterministic), and (2) because object detection with the provided code is imperfect, we expect that not all objects in the simulated radar returns will be identified correctly.

4. Code structure

Coming soon!

5. MilSpec development notes, errata, changelog

v1.1:
- Ported and verified on Ubuntu 18.04

v1.0:
- Changes from baseline SAR code for MilSpec:
    - Ported from Unified Parallel C (UPC) to standard C99
    - Modified test parameters in `param_file.txt` to increase computational requirements and task difficulty
    - Modified code and parameters to ensure deterministic test scenario: specify and save constant input dataset (simulated radar returns) to reuse across benchmark runs, initialize random seed to constant value
    - Edited/simplified performance metric display at benchmark completion

6. Acknowledgements

MilSpec is a project under development at the University of Michigan – Ann Arbor by Pete Ehrett, John Paul Koenig, Pranav Srinivasan, Todd Austin, and Valeria Bertacco. This project is supported by the Applications Driving Architectures Center, one of six centers of JUMP, a Semiconductor Research Corporation program co-sponsored by DARPA.

The original SAR implementation upon which the MilSpec version is based is available at http://www.sarbenchmark.com/. Its developers are listed in the `AUTHORS` file included with the benchmark code.