14 July 2010

Using GPU VSİPL & CUDA to Accelerate RF Clutter Simulation

2010 Symposium on Application Accelerators in High Performance Computing

Dan Campbell, Mark McCans, Mike Davis, Mike Brinkmann
dan.campbell@gtri.gatech.edu
Outline

- RF Clutter Simulation
- Validation Approach
- GPU VSIPL
- Precision Issues
- VSIPL Port, Optimization, and Results
Outline

- RF Clutter Simulation
- Validation Approach
- GPU VSIPL
- Precision Issues
- VSIPL Port, Optimization, and Results
Radar will observe echo from object…

...as well as a strong return from the ground.

Strong returns from the ground, called “clutter”, often limit the performance of radars in air-to-air and air-to-ground operations.
Synthetic Air-to-Air Clutter

7,500 Hz

10,000 Hz

12,500 Hz

Targets at same range/Doppler as clutter will be obscured.
**RF Clutter Simulation**

**Approach**: Sub-divide ground into number of unresolvable clutter patches and compute contribution of each.
RF Clutter Simulation

Notional Parameters

<table>
<thead>
<tr>
<th></th>
<th>Air-to-Air</th>
<th>SAR Imaging (Air-to-Ground)</th>
<th>Our Test</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Range Bins</td>
<td>200</td>
<td>1750</td>
<td>500</td>
</tr>
<tr>
<td># of Pulses</td>
<td>128</td>
<td>3000</td>
<td>8</td>
</tr>
<tr>
<td># of Clutter Patches</td>
<td>6,800 Rng x 96 Az = 6.5 x 10^5</td>
<td>14,500 Rng x 26,812 Az = 3.8 x 10^8</td>
<td>566 rng x 52 az = 29,432</td>
</tr>
</tbody>
</table>

Computational load depends on radar parameters and collection geometry (e.g., high resolution scenarios require a large number of independent clutter patches)
RF Clutter Simulation

Algorithm:

Inputs
• Radar Parameters (waveform, antenna, etc.)
• Location of platform for each pulse

Output
• Simulated radar data cube (sample voltage for each pulse, each channel, and each range bin)

For each pulse and for each range bin…
   For each clutter patch in this range ring…
      1. Compute range, azimuth, and elevation from platform to clutter patch.
      2. Scale contribution of this clutter patch according to the radar range equation.
      3. Accumulate the contribution of this clutter patch to the simulated data cube.
Outline

- RF Clutter Simulation
- Validation Approach
- GPU VSIPL
- Precision Issues
- VSIPL Port, Optimization, and Results
Validation Needs

• Porting MATLAB ➔ C introduces changes
  • Random Number Generator
  • Double ➔ Single
  • Implementation of some functions e.g. transcendentals
  • Reordering of operations
  • Programmer Error

• Identical output too costly

• Derive acceptance criteria from expected usage needs
Validation Approach

• Modify sim to capture RNG stream from MATLAB
• Automate large number of runs for golden data
• Accelerated port optionally ingests RNG stream
• Capture port output and compare to golden data
• Acceptance Criteria:
  - $\text{CNR}_\Delta = \frac{\text{CNR}_M - \text{CNR}_T}{\text{CNR}_M} < 10^{-4}$
  - $\text{ECR} = 20 \log_{10}(\frac{\text{norm}(M(:) - T(:))}{\text{norm}(M(:))}) < -60\text{dB}$
  - $\text{ADMSE} = \text{Mean}(\|\text{fft2}(M(:)) - \text{fft2}(T(:))\|^2) < 10^{-3}$
Outline

- RF Clutter Simulation
- Validation Approach
- GPU VSIPL
- Precision Issues
- VSIPL Port, Optimization, and Results
VSIPPL

- http://www.vsipl.org
- Industry standard C API for *portable* dense linear algebra & signal processing
  - Also C++, Python
- Accelerated implementations for many platforms, primarily embedded, coprocessor-based systems
- GPU VSIPPL VSIPPL implementation that exploits Graphics Processing Units to accelerate VSIPPL applications – developed at GTRI
  - http://gpu-vsipl.gtri.gatech.edu
#define ARRAY_SIZE 50

int main (int argc, char *argv[]) {
    vsip_vview_f *v1, *v2, *v3;
    int i;

    v1 = vsip_vcreate_f (ARRAY_SIZE, VSIP_MEM_NONE);
    v2 = vsip_vcreate_f (ARRAY_SIZE, VSIP_MEM_NONE);
    v3 = vsip_vcreate_f (ARRAY_SIZE, VSIP_MEM_NONE);

    vsip_vramp_f (0, 0.1, v1);
    vsip_vramp_f (0, 1.0f/15.0f, v2);
    vsip_vramp_f (0, 0.0, v3);

    vsip_vadd_f (v1, v2, v3);

    for (i=0; i<ARRAY_SIZE; i++)
    {
        printf ("out[%i]:%f\n", i, vsip_vget_f(v3, i));
    }
}
**Application Example: Range Doppler Map**

- Simple Range/Doppler data visualization application demo
- Intro app for new VSIPL programmer
- 431x Speedup TASP → GPU-VSIPL
- No changes to source code

<table>
<thead>
<tr>
<th>Section</th>
<th>GT200 Time (ms)</th>
<th>Q6600 Time (ms)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admit</td>
<td>0.74</td>
<td>0.0</td>
<td>0</td>
</tr>
<tr>
<td>Baseband</td>
<td>1.46</td>
<td>1872.3</td>
<td>1280</td>
</tr>
<tr>
<td>Zeropad</td>
<td>1.91</td>
<td>110.7</td>
<td>58</td>
</tr>
<tr>
<td>Fast time FFT</td>
<td>1.38</td>
<td>5696.3</td>
<td>4130</td>
</tr>
<tr>
<td>Multiply</td>
<td>13.5</td>
<td>33.9</td>
<td>2.51</td>
</tr>
<tr>
<td>Fast Time FFT⁻¹</td>
<td>1.44</td>
<td>5729.0</td>
<td>3980</td>
</tr>
<tr>
<td>Slow time FFT, 2x CT</td>
<td>9.41</td>
<td>3387.0</td>
<td>360</td>
</tr>
<tr>
<td>log₁₀</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Release</td>
<td>9.44</td>
<td>0.0</td>
<td>0</td>
</tr>
<tr>
<td>Total:</td>
<td>40.7</td>
<td>17299.42</td>
<td>431</td>
</tr>
</tbody>
</table>
Outline

- RF Clutter Simulation
- Validation Approach
- GPU VSIPL
- Precision Issues
- VSIPL Port, Optimization, and Results
### Original Validation Results

**VSIPL versions compared to MATLAB version**

<table>
<thead>
<tr>
<th></th>
<th>VSIPL Double</th>
<th>VSIPL Single</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNR Consistent</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>CNR $\Delta$</td>
<td>$10^{-1.6}$</td>
<td>$10^{-6}$</td>
<td>$&lt; 10^{-4}$</td>
</tr>
<tr>
<td>ECR</td>
<td>-152 dB</td>
<td>2.9 dB</td>
<td>$&lt; -60$ dB</td>
</tr>
<tr>
<td>ADMSE</td>
<td>$10^{-1.2}$</td>
<td>$10^{4}$</td>
<td>$&lt; 10^{-3}$</td>
</tr>
</tbody>
</table>
Single Precision

- Single precision errors caused by high dynamic range in platform to clutter patch range calculation:
  - \( d(\text{Platform} \rightarrow \text{clutter}) \gg d(\text{clutter patch} \rightarrow \text{clutter patch}) \)

- Solution: use far-field approximation technique
  - Double precision used to compute a base range
  - Single precision for sets of \( \Delta R \) values
  - Small number of double precision calculations has negligible affect on performance
Far Field Approx. via Taylor Expansion

Range between platform at $x$ and clutter patch at $y$

\[ R(x) = \| x - y \| \]

Linear approximation near $x_0$

\[ R(x) \approx R(x_0) + \left( \frac{x_0 - y}{\| x_0 - y \|} \right) \cdot (x - x_0) \]

- Unit vector from CPI center to clutter patch
- Distance from center of scene, $\epsilon$
- Distance travelled in direction orthogonal to “lines” of constant range

Quadratic Term

\[ \frac{1}{2} \left[ \frac{\| \epsilon \|^2}{\| x_0 - y \|} - \left( \epsilon \cdot \frac{x_0 - y}{\| x_0 - y \|} \right)^2 \right] \approx 0 \text{ for } \| \epsilon \| \ll \| x_0 - y \| \]
Bounding Error

Approximation Error

\[ R(x_0 + \epsilon) - \hat{R}(x_0 + \epsilon) \leq \frac{1}{2} \frac{||\epsilon||^2}{||x_0 - y||} \]

Case 1: Air-to-Air

128 pulses, 20 kHz PRF, 300 m/s velocity → ||\epsilon|| < 1m
10 km Altitude → ||x_0 - y|| < 10km
error < 50 \mu m < 0.06° phase at X band

Case 2: SAR

10 second dwell, 100 m/s velocity → ||\epsilon|| < 500m
10 km Altitude → ||x_0 - y|| < 10km
error < 12.5 m >> \lambda at X band!!!

Linear approximation to range may be appropriate for typical air-to-air scenarios.
## Validation Results

### Comparison to original MATLAB version
- Approximation technique used in each version listed

<table>
<thead>
<tr>
<th></th>
<th>MATLAB Single</th>
<th>VSIPL Double</th>
<th>VSIPL Single</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CNR Consistent</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><strong>CNR Δ</strong></td>
<td>$10^{-7}$</td>
<td>$10^{-14}$</td>
<td>$10^{-5}$</td>
<td>$&lt;10^{-4}$</td>
</tr>
<tr>
<td><strong>ECR</strong></td>
<td>-101 dB</td>
<td>-130 dB</td>
<td>-98 dB</td>
<td>$&lt;-60$ dB</td>
</tr>
<tr>
<td><strong>ADMSE</strong></td>
<td>$10^{-7}$</td>
<td>$10^{-10}$</td>
<td>$10^{-6}$</td>
<td>$&lt;10^{-3}$</td>
</tr>
</tbody>
</table>
Outline

- RF Clutter Simulation
- Validation Approach
- GPU VSIPL
- Precision Issues

VSIPL Port, Optimization, and Results
VSIPL PORT

- MATLAB to VSIPL port made easier due to VSIPL functions that emulate MATLAB operations
- Original MATLAB code very complex, particularly for radar novice
  - First pass of the port was done with almost no attempts at optimizations
- GPU transitioned required some additional changes
  - Single vs Double precision issues
  - Time cost of operations differ TASP ↔ GPU
Optimization Issues

• MATLAB code written for readability over speed
  • Too many nested loops, operations involving small datasets
  • Many redundant calculations

• Original code was very flexible, due to large user base
  • Most optimizations required removing some generality
  • Assumptions need to be made about the scenario

• Abstraction barrier issues
  • Small operations less costly on CPU than GPU
  • Operation fusion, coarser operations, and leaving small things in C each helped
HPC Port – Performance

- Optimization progression of single precision VSIPL:

  - Reduced generality; Dynamic ➔ Static
  - Small ops VSIPL ➔ C
  - Reduced generality; simplified operations
  - Hoisted invariants; reordered for fusion
  - Stride consciousness; coarser VSIPL ops; loop fusion
HPC Port – Performance

- Performance Timing Results:

<table>
<thead>
<tr>
<th>Version</th>
<th>Runtime(s)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATLAB</td>
<td>162.5</td>
<td>1x</td>
</tr>
<tr>
<td>TASP VSIPL Double</td>
<td>20.9</td>
<td>7.8x</td>
</tr>
<tr>
<td>TASP VSIPL Single</td>
<td>14.0</td>
<td>11.6x</td>
</tr>
<tr>
<td>GPU VSIPL Single</td>
<td>2.2</td>
<td>73.8x</td>
</tr>
<tr>
<td>CUDA Native</td>
<td>TBD</td>
<td></td>
</tr>
</tbody>
</table>

- GTX 280/Q6600 TASP single core only