A FAST ULTRASONIC SIMULATION TOOL BASED ON MASSIVELY PARALLEL IMPLEMENTATIONS

Jason LAMBERT¹, Gilles ROUGERON¹, Lionel LACASSAGNE², and Sylvain CHATILLON¹.

¹CEA, LIST, F-91191 Gif-sur-Yvette, France
jason.lambert@cea.fr

²LRI, Université Paris-Sud, F-91405 Orsay cedex, France
OUTLINE

Context and objectives

Description of the UT simulation tools

• Beam computation
• Defect diffraction
• Algorithm of a simulation

Description of CPU-multicore and GPU implementations

• CPU and GPU specificities
• Model validation with CIVA 11
• Performances results

Conclusions and perspectives
**Objectives**: fast UT inspection simulation

- Inspection designing tools (probe especially phased array, scanning, …)
- POD curves computation, Data Base computation, inversion procedure

**Means**: exploitation of CPU-multicores and GPU capabilities

- Intensively parallel computation (beam propagation and diffraction with defect)
- Simplified model based on analytical solution (beam propagation)
- Use of high performances signal processing libraries (Intel MKL/NVidia CUFFT)

**Limitations**: canonical configurations

- Simple geometries described by planar surfaces (planar, 2D CAD)
- Homogeneous and isotropic materials
- Planar defects (Kirchhoff approximation)
- Half-skip
DESCRIPTION OF THE UT SIMULATION TOOLS
Simplified model for isotropic and homogeneous planar component

1 - **Polynomial** for ray paths computation (direct and after backwall reflection) solved using Newton method

- Ray path optimization (no ray tracing)
- Regular Probe discretization

2 - Analytical expression of the **amplitude of the contribution** (no matrix computation)

- Divergence factor
- Reflection and refraction coefficients

**Computation of the impulse response**

\[ h(M,t) \]

**Superposition of elementary contributions over the probe**

\[ u(M,t) = h(M,t) \otimes v(t) \]

**Observation point**

**Source point**

**Extraction of beam characteristics** for echo simulation

- Amplitude, phase shift and time of flight
- Direction of propagation and polarization
**Generic formulation**: Transmission Beam / Flaw scattering / Reception beam \((by\ Auld’s\ reciprocity)\) over the flaw discretization

- **Incident beam**
  - Amplitude: \(q_\text{E}(M)\)
  - Time of flight: \(T_\text{E}\)
  - Incident direction
  - Incident polarization

- **Received beam**
  - Amplitude: \(q_\text{R}(M)\)
  - Time of flight: \(T_\text{R}\)
  - Observation direction
  - Observation polarization

**Synthesis of the received signal using Auld’s reciprocity principle**

\[
S_{ER}(t) = \sum_{M} q_\text{R}(M)q_\text{E}(M)\{K(M,t).e^{i\phi_\text{E}}.e^{i\phi_\text{R}}\}.S(t-T_\text{E}-T_\text{R})
\]

- **Flaw meshing**
- **Kirchhoff diffraction coefficient, valid near specular reflection**

\[K(M,t)\]
ALGORITHM OF A DIFFRACTION SIMULATION

START

Field impulse response
- Ray path computation
- Amplitude & time of flight

Sum of elementary contributions following delay laws:
Sum of displacement(t) along x,y,z coordinate
\[ \text{displacement}(t) = \{x(t), y(t), z(t) \} \]

Extraction of beam characteristic:
- \[ \text{dir} / \|\text{dir}\| = \max \|\text{displacement}(t)\| \]
- \[ A(t) = \text{dir} \cdot \text{displacement}(t) \]
- Amplitude, time of flight & phase shift extraction: maximum of the enveloppe of (projection signal)
\[ A = A_{\text{max}} \cdot e^{i\varphi} \]

Echo Impulse Response computation
- Bilinear Interpolation of beam characteristics from a coarse field grid (\(\lambda\))
- Kirchhoff coefficients

Echo signal formation by convolution

END
DESCRIPTION OF THE CPU-MULTICORE AND GPU IMPLEMENTATIONS
OVERVIEW OF CPU AND GPU PLATFORM

Multicore CPU

- Small number of cores (10s)
- Cores can work on different tasks (task parallelism)
- Core can handle complex task

GPU

- Numerous smaller cores (100s)
- A same task executed on multiple data by multiple cores (data parallelism)
- Memory access shared across cores
- Coprocessor: separate memory from host CPU
Algorithm optimized for coarse grained parallelism

- High level loops are multithreaded with OpenMP API at each step
- Memory quite abundant on CPU so results are reused (field impulse responses)

Signal processing use Intel Math Kernel Library (MKL)

- Need to evaluate the time gate of signals to initialize only once FFT operations
- Calls to FFT are made by threads when needed, without any synchronization
GPU CONSIDERATIONS

GPU are aimed at data parallel computations

- Promote analytical and specialized approach over genericity
- Control flow optimized for executing the same task over multiple data, in subroutines called “kernels”
- Benefits from GPU efficient collaboration of nearby threads (shared memory)
- cuFFT library optimized for batches of multiple signals

GPU are coprocessor driven by their host CPU

- Memory allocation should be done prior to executing GPU kernels
- Memory management, kernel launches and cuFFT calls are made asynchronously by the host
- GPU have less device memory than CPU have commonly
Signal size determination

- Time of flight computation model using analytical ray path (done for the Total Focusing Method [1])
- Size is sent back to host to allocate memory on the GPU

UT Field computation – summation of probe elementary contributions

- Contributions are recomputed, cheaper than storing and retrieving from memory
- Signals resulting from the summation too big to fit in GPU memory
  → work divided along transducer positions and temporary data will be overwritten

Defect response computation by Kirchhoff Model

- Computed once on the whole dataset (every positions & delay law)
- Synthetic inputs (direction, amplitude & time of flight) will fit in the GPU memory
- Allows cuFFT to work on a batch of signals for the last convolution
- Size of resulting echo signal given by the user in input data

[1] Implementation of a GPU Accelerated Total Focusing Reconstruction Method within Civa Software
---Gilles Rougeron, Jason Lambert, and Ekaterina Iakovleva, Lionel Lacassagne,
QNDE 2013, Session 28
Flat bottom holes:

- Planar surface, L direct echo, homogeneous isotropic material (steel)
- Series of 10 Ø 3mm FBH
- 1 out of 7 delay law: delay law L45 + 25mm depth focusing
- Reference: CIVA, validated against the benchmark

CIVA
CPU
GPU

Time of flight OK
Flat bottom holes:

- Planar surface, L direct echo, homogeneous isotropic material (steel)
- Series of 6 Ø3mm FBH
- 1 out of 7 delay law: delay law L45 + 20 mm depth focusing

0.8dB
0.6dB
0.6dB

CIVA
CPU
GPU

Time of flight OK
Planar defects (backwall breaking notches)

- Planar surface, homogeneous isotropic material (steel)
- Current implementation: L waves only
- L corner echo on 15 mm height notch
- 1 out of 4 delay law: delay law L45 + 30mm depth focusing

Time of flight OK
### Dataset Breakdown

<table>
<thead>
<tr>
<th>Dataset</th>
<th>FBH - Series 1</th>
<th>FBH – Series 2</th>
<th>Notch</th>
</tr>
</thead>
<tbody>
<tr>
<td># of defects</td>
<td>10 FBH</td>
<td>6 FDH</td>
<td>1 notch</td>
</tr>
<tr>
<td>Probe</td>
<td>64 channels ↔ 128 samples</td>
<td></td>
<td></td>
</tr>
<tr>
<td># of positions</td>
<td>361</td>
<td>201</td>
<td></td>
</tr>
<tr>
<td># delay laws</td>
<td>7</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td># UT field sampling</td>
<td>90</td>
<td>54</td>
<td>100</td>
</tr>
<tr>
<td># defects sampling</td>
<td>4140</td>
<td>2484</td>
<td>12321</td>
</tr>
<tr>
<td>Mode</td>
<td>L direct</td>
<td>L direct</td>
<td>Half skip – L</td>
</tr>
</tbody>
</table>

**FBH**: Fatigue Block Heel

**FDH**: Fatigue Damage Heel

**UT**: Ultrasonic Testing
PERFORMANCES – MULTICORE CPU VERSION

Hardware configuration

- 2x CPU Intel Xeon 5590
  6 cores @3.47Ghz each + Hyperthreading (2x12 logical cores)
- 24 GB of memory

CPU implementation specificities

- Use optimized Intel Math Kernel Library for FFT computations
- Each step is parallelized with OpenMP API
- Simple precision computations (IEEE 754)


- This machine is also the one running CIVA 11.0
## PERFORMANCES – MULTICORE CPU

<table>
<thead>
<tr>
<th>Dataset</th>
<th>FBH - Series 1</th>
<th>FBH – Series 2</th>
<th>Notch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fast model on CPU</td>
<td>7,6 s</td>
<td>2 s</td>
<td>7,3 s</td>
</tr>
<tr>
<td>CIVA 11.0</td>
<td>241 s</td>
<td>142 s</td>
<td>363 s</td>
</tr>
<tr>
<td>Gain / CIVA 11.0</td>
<td>x31,7</td>
<td>x71</td>
<td>x49,7</td>
</tr>
</tbody>
</table>

Signal processing operations remain the most costly, especially for one echo mode.
PERFORMANCES – GPU

Hardware configuration

- NVidia GeForce GTX580 (high end, consumer grade GPU)
  - 512 CUDA cores @1.544Ghz each
  - Fermi architecture
- 1,5 GB of device memory

GPU implementation specificities

- CUDA toolkit version 5.0 on Windows
- Simple precision computations (IEEE 754)
- Execution time do not include GPU initialization nor the transfer of the result from the device back to the host (at most 200ms on the studied cases)
### PERFORMANCES – GPU

<table>
<thead>
<tr>
<th>Dataset</th>
<th>FBH - Series 1</th>
<th>FBH – Series 2</th>
<th>Notche</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPU version</td>
<td>5,8 s</td>
<td>1,7 s</td>
<td>3,6 s</td>
</tr>
<tr>
<td>CPU version</td>
<td>7,6 s</td>
<td>2 s</td>
<td>7,3 s</td>
</tr>
<tr>
<td>Gain</td>
<td>×1,2</td>
<td>×1,17</td>
<td>×2</td>
</tr>
</tbody>
</table>

Signal processing is costly:
- FFT over big memory chunks
- Summation require Atomic Operation

Signal processing operations remain the most costly, especially for one echo mode.
CONCLUSIONS

• A fast UT inspection simulation tool has been developed.

• Implementation on multi-core CPU and GPU

• Validated on QNDE benchmark 2013 (comparison with CIVA)

• Performances: Acceleration ranging on real data
  • CPU (12 cores + hyperthreading): from \( x_{31} \) to \( x_{70} \) over CIVA 11
  • GPU (512 cores): from \( x_{1.17} \) to \( x_{2} \) over CPU (12 core + HT) 
  from \( x_{41} \) to \( x_{101} \) over CIVA 11
- Extension of CPU/GPU implementations:
  - Another type of defect (e.g. SDH)
  - Another type of diffraction model (e.g. GTD, SOV)
  - On the GPU, T mode computation (C computations requiring 2x the memory)

- Evolution of the CPU implementation to use SIMD instruction to benefit from short vector operations

- Reducing the number of signal processing operations using CIVA model expertise

- Integrated within CIVA 12

- Interactive real-time simulation is within reach (Sectorial Scan, …)
Thank you for your attention

Sylvain.chatillon@cea.fr
Partially supported by ANR (OPARUS Projet)