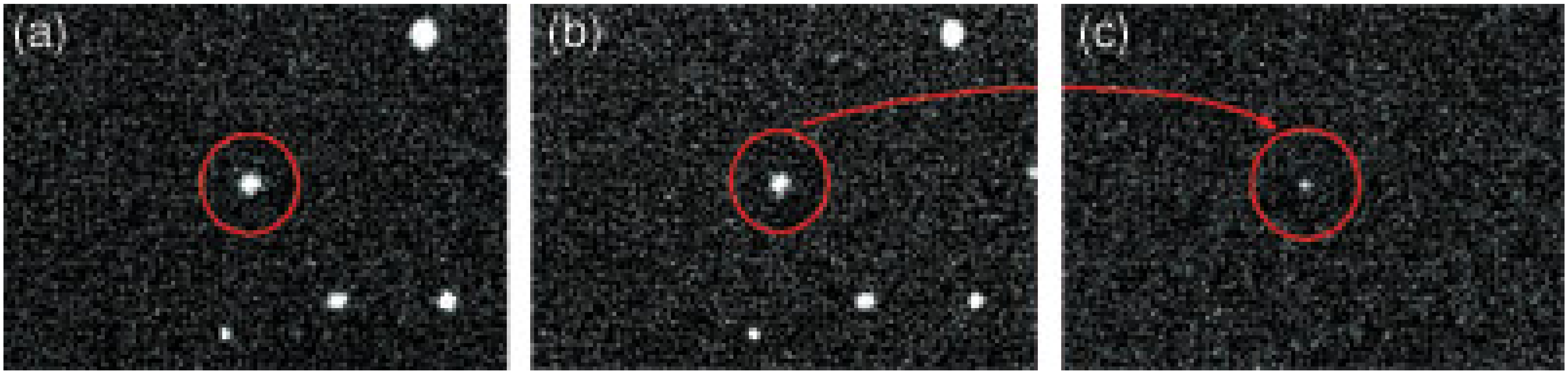
Heterogeneous Parallel Programming COMP4901D

Parallel Astronomical Image Subtraction

Overview

- Astronomical image subtraction
- Existing CPU-based software
- Our parallelization on the CPU and the GPU

Astronomical Image Subtraction



(Resource: A Wide New Window on the Universe, S&TR November 2005)

- C. Alard, *Image Subtraction Using A Space-varying Kernel*, Astronomy & Astrophysics Supplement Series, 2000 June.
- C. Alard, RH. Lupton, *A Method for Optimal Image Subtraction*, The Astrophysical Journal, 1998 August .

Pixel-By-Pixel Subtraction?

- Different image sizes
- Different telescopes
- Changes in atmospheric conditions
- Changes in optical fluctuations

The Optical Image Subtraction Method

- Foundation
  - The point spread functions (PSF) of optical images are temporally invariant transfer functions.
- Idea
  - First compute a convolution kernel to match the PSFs of two astronomical images
  - Then use the kernel to generate an output image that yields a constant field except some significant deviations that reflect the variations in the brightness or spatial locations of the celestial objects.

Output Image Computation

- Goal: $O(x, y) = T(x, y) \otimes K(u, v) - I(x, y)$, where T- template image, I – input image, O – output image, K – kernel, (x,y) - pixel location in image, (u,v) – pixel location in kernel box.
- K(u,v) is a kernel that minimizes

$$\sum(T(x, y) \otimes K(u, v) - I(x, y))^2.$$

Kernel Computation

$$K(u,v) = \sum_{n=1}^{N} a_n(x,y) K_n$$

$$K_n(u,v) = e^{-\left(u^2+v^2\right)/2\sigma_k^2} u^i v^j$$

$$Ma = B$$

$$M_{ij} = \int \left[T \otimes K_i\right](x,y) \frac{\left[T \otimes K_j\right](x,y)}{\sigma(x,y)^2} dxdy$$

$$B_i = \int I(x,y) \frac{\left[T \otimes K_i\right](x,y)}{\sigma(x,y)^2} dxdy$$

HOTPANTS (High Order Transformation of Psf and Template Subtraction)

- An open-source software package developed by Andrew Becker of Univ. of Washington
- Follows the OIS method [Alard 2000]
- Sequential C program
- Widely used in astronomy community

http://www.astro.washington.edu/users/becker/v2.0/hotpants.html

HOTPANTS Workflow

- Initialization
  - Read input image & template; allocate memory; Compute Kn
- Filling
  - Compute the repetitive part in Mij and Bi
- Checking
  - sigma-clip the outlier; rebuild the kernel
- Convolution
  - Compute the space-varying kernel; perform the convolution
- Output

HOTPANTS Time Breakdown

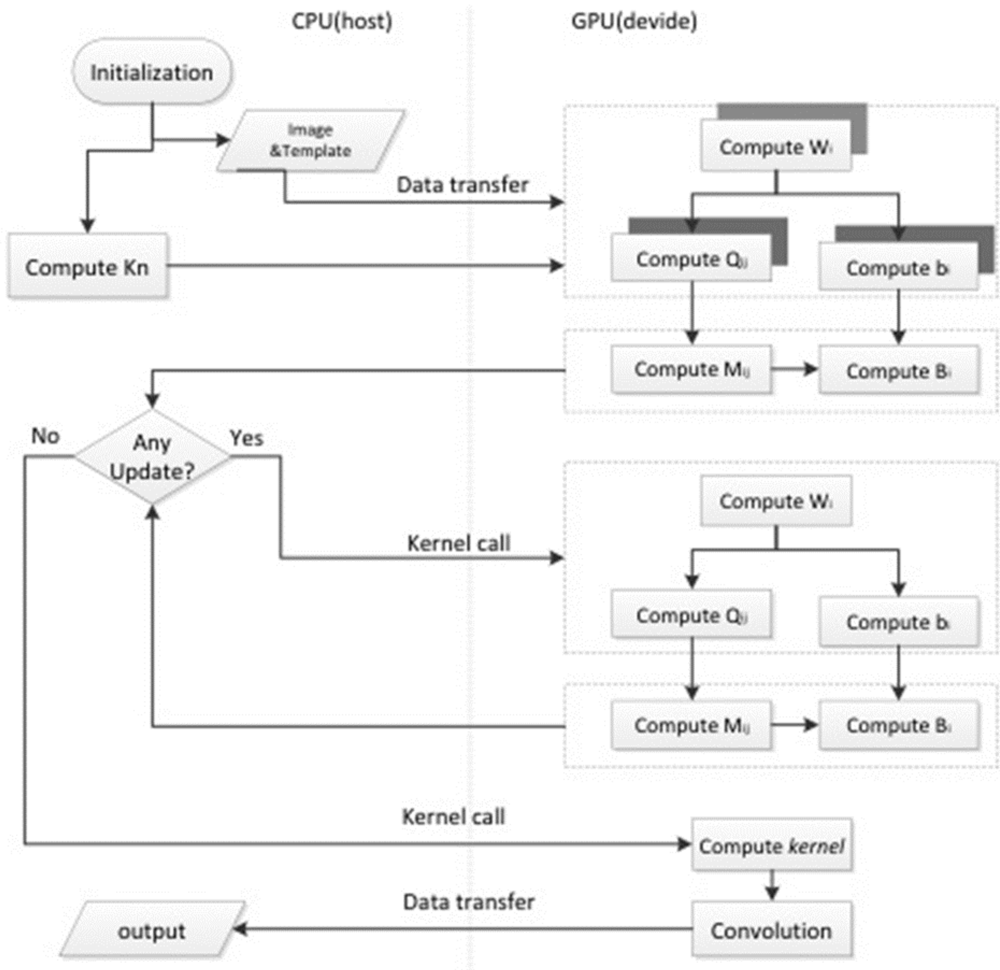
- The convolution step takes 2/3 of the total running time.
- Filling and Output together takes 1/5 of the overall time.
- Initialization and Checking together 1/10.

Step	Init.	Fill	Check	Conv.	Output	Total
Time(seconds)	0.9	1.52	0.38	7.96	1.04	11.8
Percentage(%)	7.6	12.9	3.2	67.5	8.8	100

Parallelizing HOTPANTS

- Initialization – keep on the CPU
  - Mostly sequential; exponential functions
- Filling – transfer and compute on the GPU
  - Parallel computation on stamps of image
- Checking – checking on CPU; compute on GPU
  - Checking involves branching; only a flag on CPU
- Convolution – compute on GPU
- Output – transfer to CPU

P-HOTPANTS Workflow



Implementation Issues

- Number of dimensions of the GPU kernel program
  - Each block of GPU threads is responsible for computing on the pixels of a sub-image.
  - The default size of a sub-image in HOTPANTS is 21 x 21.
  - Configuring the kernel to 2D 21X21 blocks causes 34% waste of threads (32-thread warp scheduling)
  - Uses 1D thread blocks: better thread utilization

Implementation Issues (cont.)

- Size of data partitioning
  - Partitions the image data for blocks of threads
  - Too big a partition: registers and shared memory assigned insufficient for each thread's data.
  - Too small a partition: too little work for each thread
  - Experimentally tuning the partition size

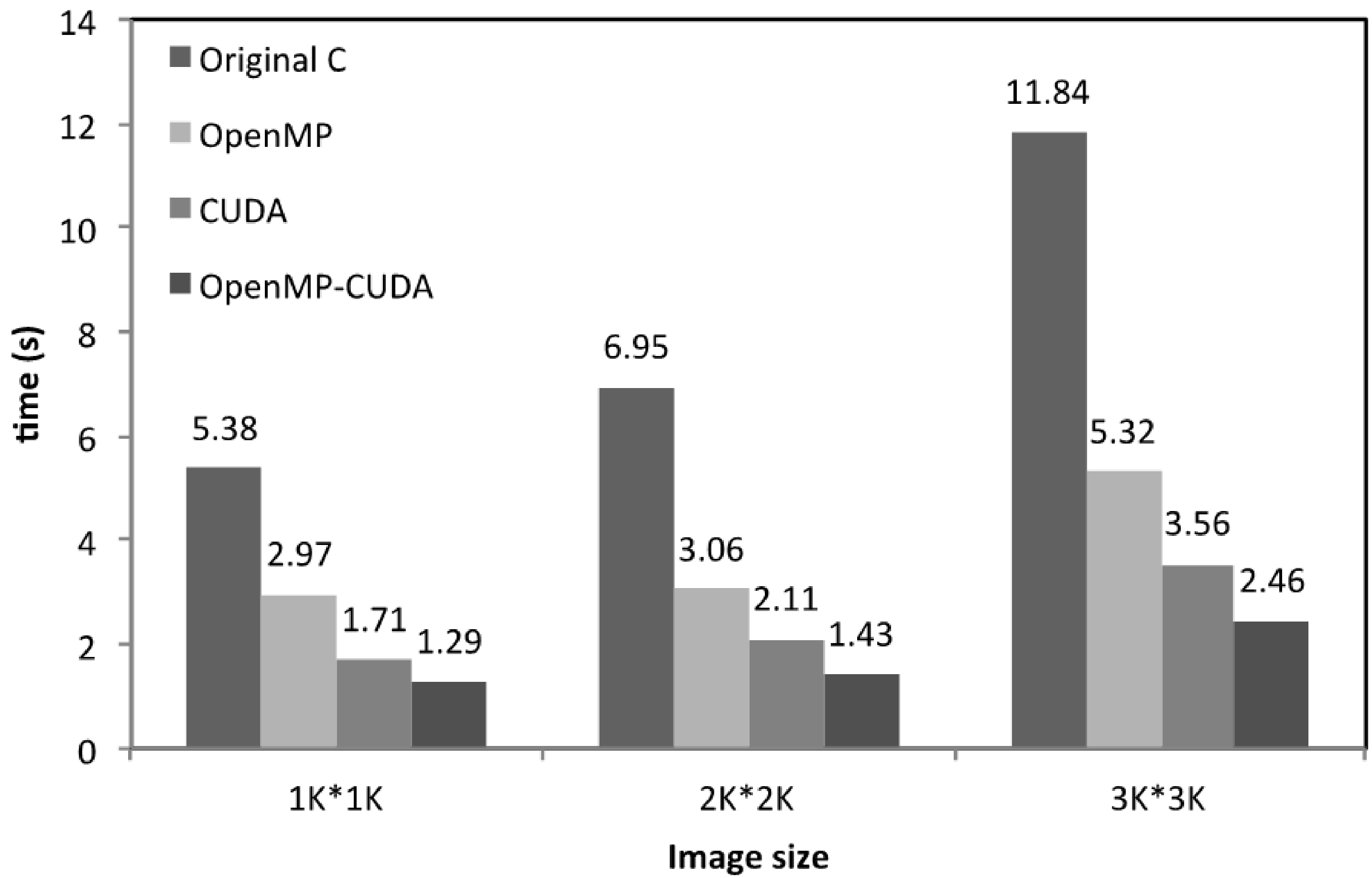
Implementation Issues (cont.)

- Use of the GPU memory hierarchy
  - Tiling; use shared memory; reuse global memory
- Combining multiple GPU kernel programs
  - Only a few with the same thread block size
- Computation Order
  - Assignment of inner and outer loops
- GPU-CPU co-processing
  - CPU has both sequential and parallel tasks

Experimental Setup

- Desktop computer
  - Intel i7-2600K quad-core CPU, NVIDIA GTX580GPU
  - 16 GB main memory, 3GB device memory
  - PCE-e bus transfer at 5GB/second
- Software
  - Scientific Linux 6.2, gcc 4.4.5 and nvcc V0.2.1221.
  - The original HOTPANTS 5.1 with default setting,
  - OpenMP-HOTPANTS (OpenMP on CPU),
  - CUDA-HOTPANTS (CUDA on GPU), and
  - P-HOTPANTS (OpenMP on CPU and CUDA on GPU).

Overall Performance



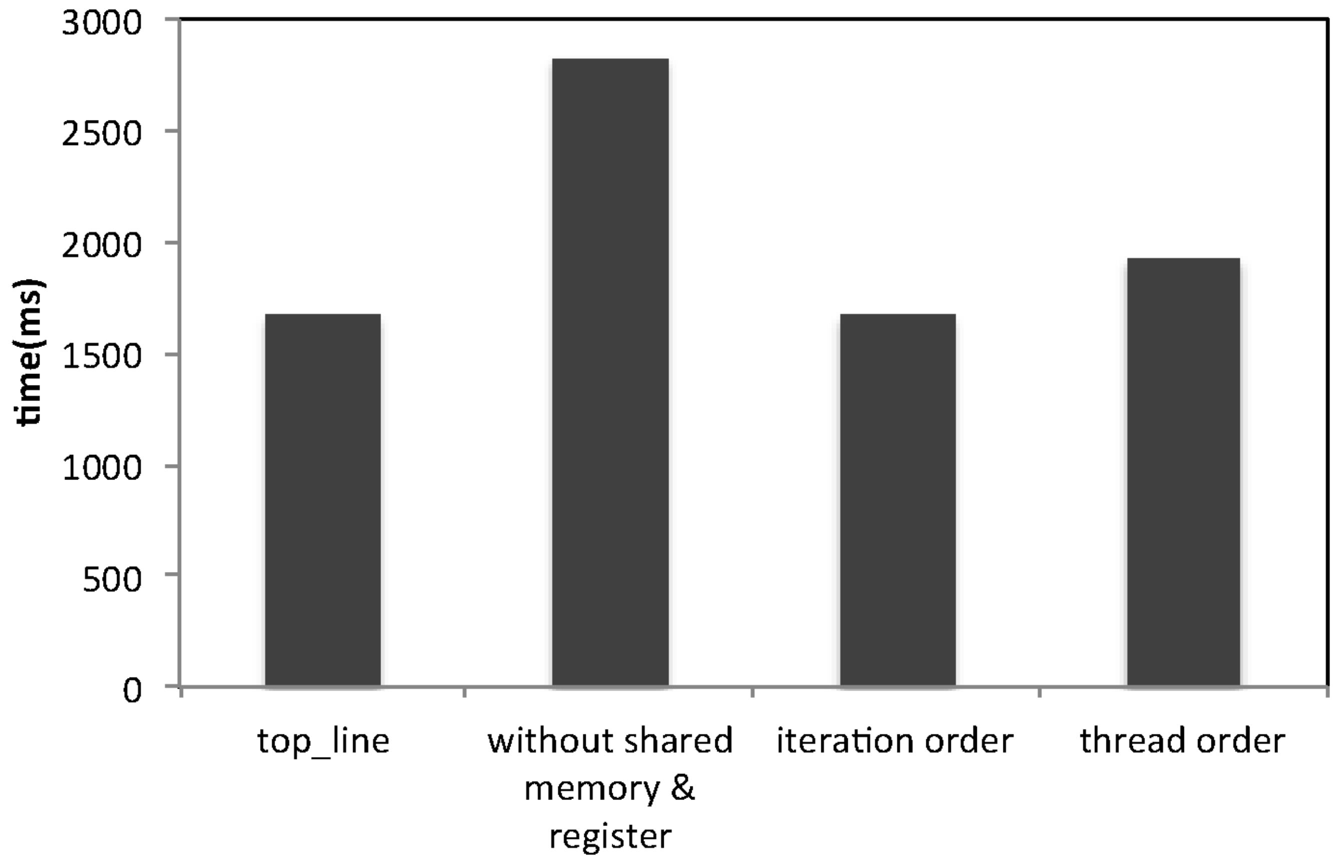
Time Breakdown

Step	Init.	Fill	Check	Conv.	Output	Total
Original	0.90	1.52	0.38	7.98	1.05	11.84
(seconds)	7.5%	12.9%	3.2%	67.5%	8.8%	–
OpenMP	0.40	1.33	0.33	2.56	0.70	5.32
(seconds)	7.5%	25.0%	6.2%	48.1%	13.1%	–
Speedup	2.25	1.14	1.15	3.08	1.5	2.22
CUDA	0.90	0.17	0.14	1.36	0.98	3.56
(seconds)	25.5%	4.8%	3.9%	38.2%	27.5%	–
Speedup	1.00	8.94	2.71	5.80	1.07	3.32
P-HOTPANTS	0.41	0.17	0.14	1.04	0.70	2.46
(seconds)	16.7%	7.9%	5.7%)	42.3%	28.4%	–
Speedup	2.20	8.94	2.71	7.58	1.5	3.52

Convolution Performance

Image size	1K x 1K	2K x 2K	3K x 3K
Original (seconds)	2.01	3.76	7.77
CUDA (seconds)	0.505	0.821	1.832
OpenMP (seconds)	0.725	1.281	2.832
Estimated GPU-ratio	0.59	0.61	0.61
CUDA-OpenMP (seconds)	0.364	0.610	1.258
Speedup	5.52	6.16	6.18
Best GPU-ratio	0.70	0.75	0.70
CUDA-OpenMP (seconds)	0.292	0.469	0.988
Speedup	6.89	8.02	7.86

Performance Factors in Convolution



Summary

- Image convolution is most time-consuming component in astronomical image subtraction.
- P-HOTPANTS parallelizes HOTPANTS utilizing both the multicore CPU and the GPU.
- P-HOTPANTS is 7-8 times faster in convolution and 4-5 times faster in the overall performance than the original HOTPANTS.