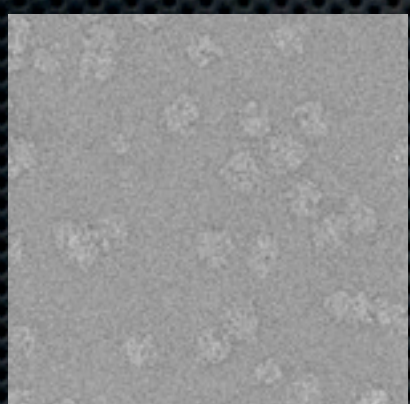
GPU Case Study: 2D Reference-based Alignment

Ryan Bubinski
Frank Lab, CUMC

August 7, 2010

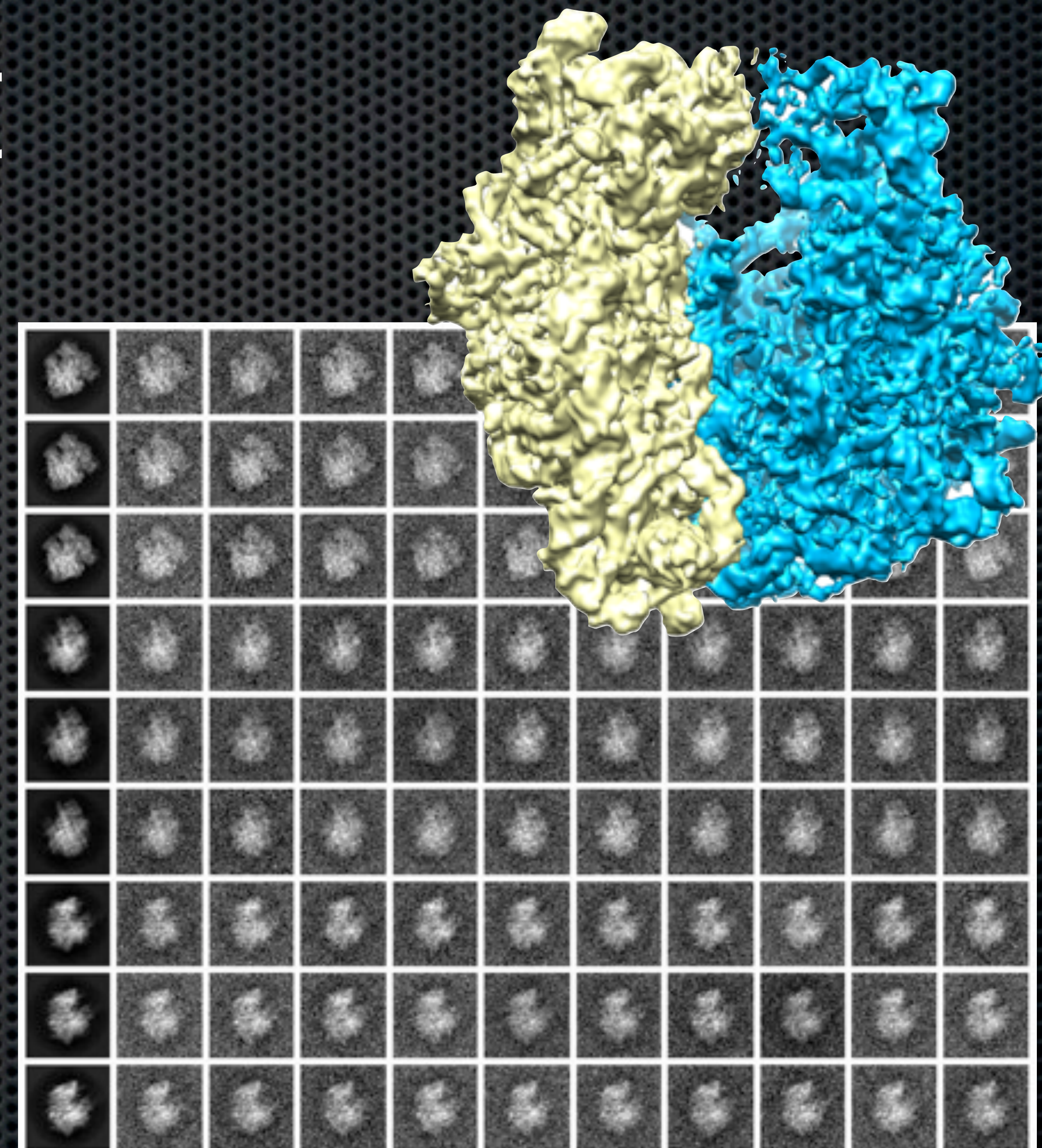
Case study: 2D reference-based alignment



Particle projections

+

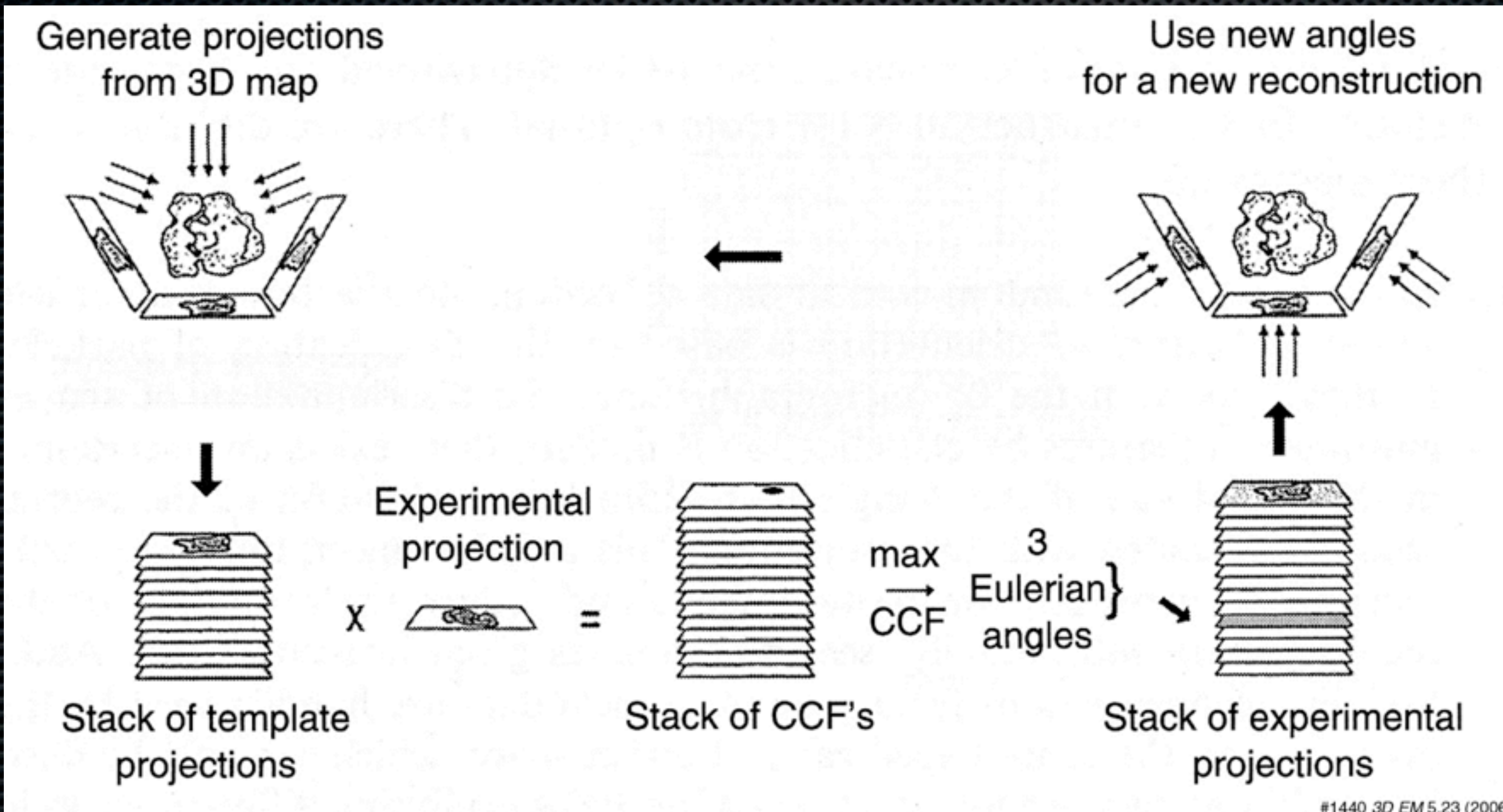
Reference projections



Particle projections aligned to Eulerian angles of reference projections

Algorithm overview

For two image sets, match each image in first set to an image in the second set.



General implementation of alignment algorithm taken from SPIDER, outlined in *Three Dimensional Electron Microscopy of Macromolecular Assemblies*, Joachim Frank 2006.

Algorithm overview

Pseudo code

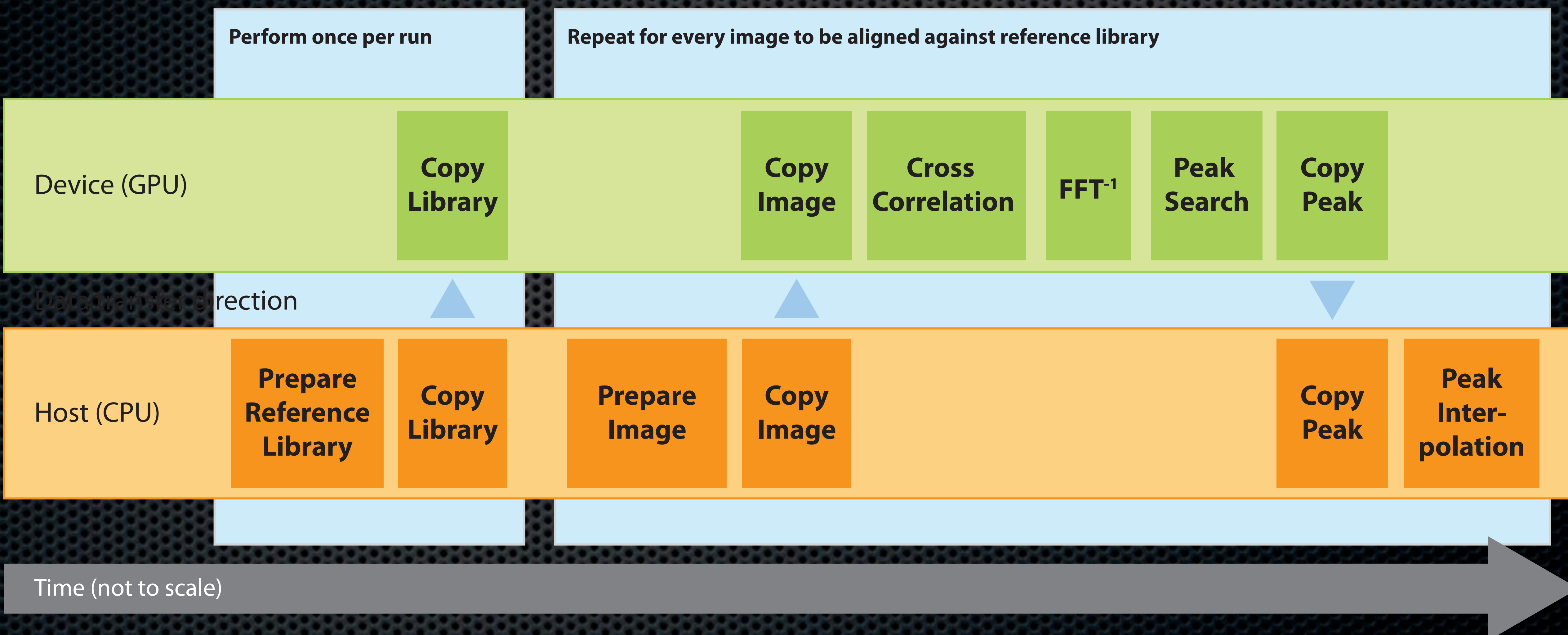
```
For each image:
  (if experimental image) translate within windowed range
  Interpolate into polar coordinates
  Normalize polar representation of image
  1D FFT each ring

For each experimental image:
  For each translation of experimental image:
    For each reference image:
      cross correlation coefficient( experimental image, reference image )
  1D FFT^-1 CCC array
  Find maximum CCC and report corresponding <reference, rotation, translation>
```

Cross correlation coefficient calculation

```
Image a, b; // complex[ rows ][ cols ]

ccc[ ring width ]; // complex[ cols ]

ccc [ ring width ] = { complex(0, 0) }; // initialize to 0

for each col:

  for each row:

    ccc[ col ] += conjugate(a[row][col]) * conjugate(b[row][col])
```

Algorithm overview

Host-device work division



Algorithm performance

Test parameters:

1 particle (100 x 256, 100 translations)

800 references (100 x 256, no translations)

Time to load particle, polar interpolate, and translate included for overall speedup (104.5ms CPU; 179.6 ms GPU)

Overall speedup: 104x

Cross-correlation

CPU time: 30380.7 (ms)

GPU time: 191.8 (ms)

Speedup: 158.4 (x)

Matrix Normalization

CPU time: 137.4 (ms)

GPU time: 7.0 (ms)

Speedup: 19.6 (x)

1D Batch FFT (CUFFT)

CPU time: 143.1 (ms)

GPU time: 2.3 (ms)

Speedup: 62.9 (x)

Peak search (CUBLAS)

CPU time: 153.3 (ms)

GPU time: 1.1 (ms)

Speedup: 136.1 (x)

Future work

- Creative use of shared memory
- Texture memory (especially in interpolation)
- Different levels of parallelization
- Benchmarking
- Automate block, grid dimensionality optimization

Algorithm overview

Particle, reference alignment

Prepare particles, references

Operations:

- Polar interpolation/ translation of images
- Normalize interpolated images

→

Most computationally expensive step

Find cross correlation coefficient (CCC) for each translated particle, reference pair

Operations:

- 1D FFT/FFT^{-1}
- CCC calculation in Fourier space

→

Find max interpolated CCC for each particle

Operations:

- Find CCC maximum
- Interpolate reported peak