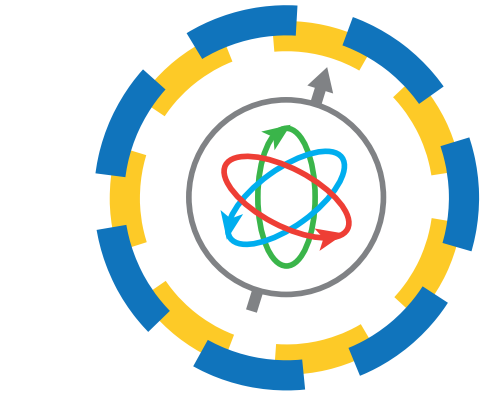
GPU accelerated reconstruction of 2D J-PET



J-PET

Adam Strzelecki,
Jakub Kowal, Piotr Białas

Jagiellonian University, Poland

September 23, 2014

Goals

- Near-realtime reconstruction
- Run on consumer hardware
- Embedded in the detector

Goals

- Near-realtime reconstruction
  - 1 sec for single iteration
  - 1 min for reconstruction
- Run on consumer hardware
- Embedded in the detector

Goals

- Near-realtime reconstruction
- Run on consumer hardware
  - consumer GPU i.e. *GeForce GTX 770*
  - *GNU/Linux* operating system
- Embedded in the detector

Goals

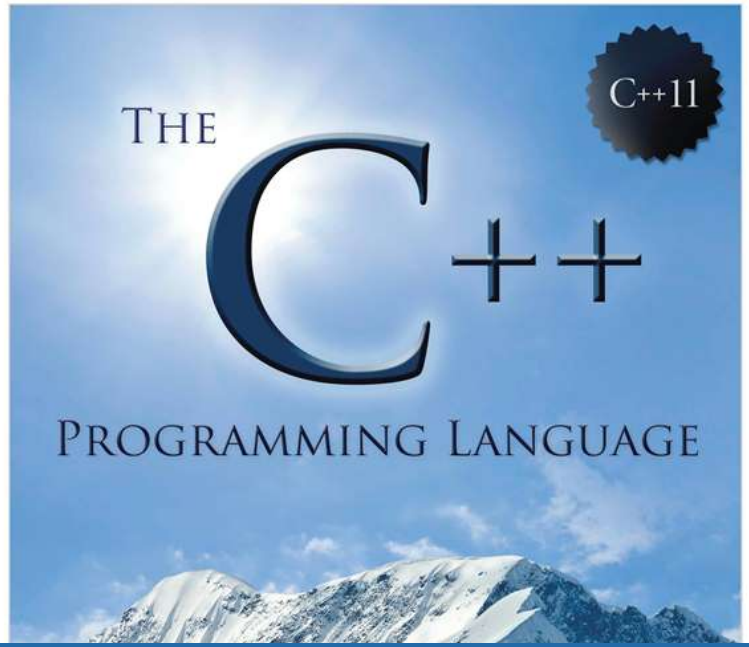
- Near-realtime reconstruction
- Run on consumer hardware
- Embedded in the detector
  - normal size PC

What we have

- C++11 generic CPU implementation for 2D x/y and z/y axis
- CUDA GPU accelerated implementation
- Xeon Phi implementation *in the works*

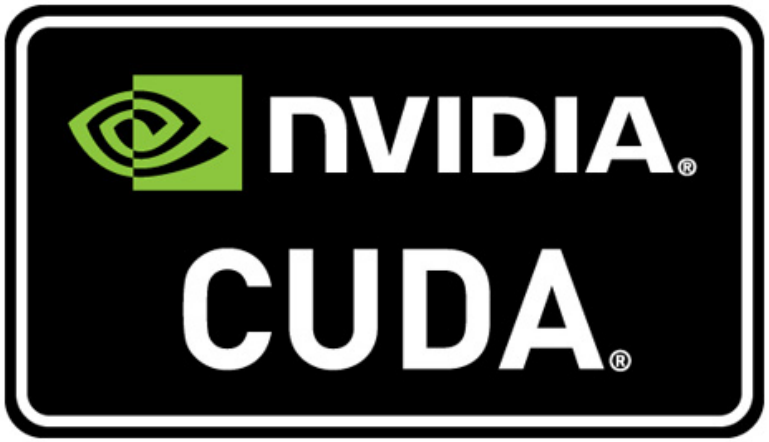
What we have

- C++11 generic CPU implementation for 2D x/y and z/y axis
  - compiles on *Linux*, *OS X* and *Windows*
  - employs *CMake* for build process
  - auto-tests via *Catch*
- CUDA GPU accelerated implementation
- Xeon Phi implementation *in the works*



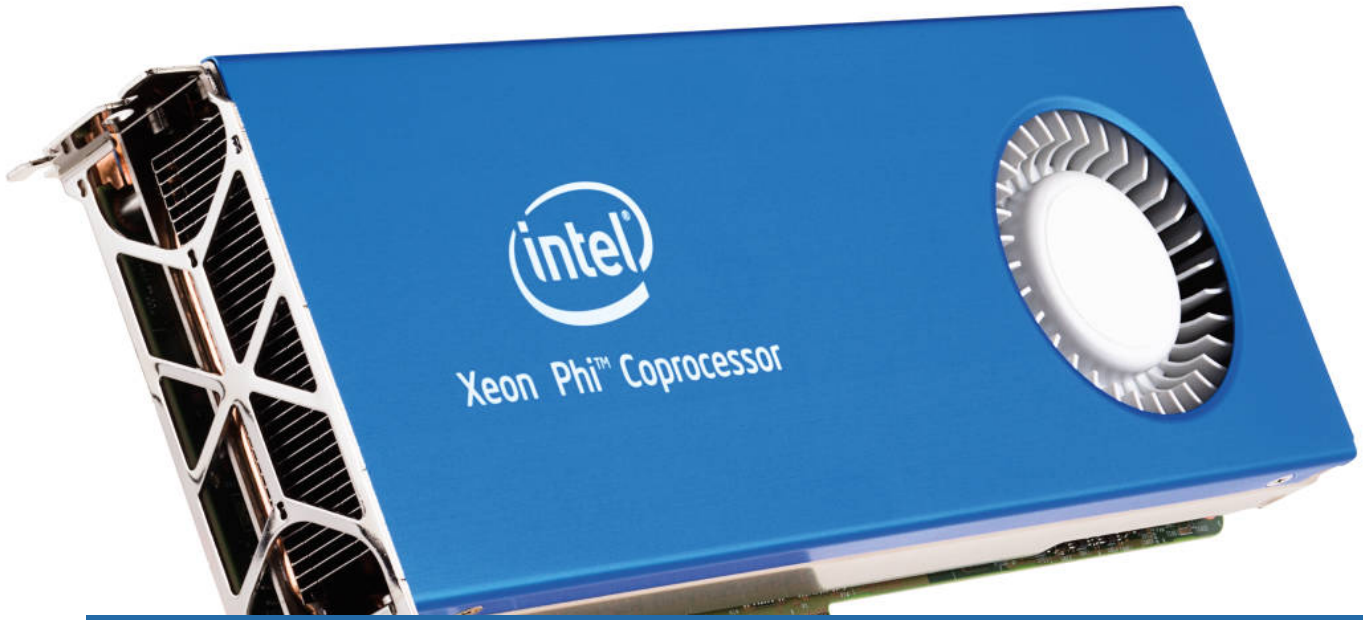
What we have

- C++11 generic CPU implementation for 2D x/y and z/y axis
- CUDA GPU accelerated implementation
  - Most of code is shared with GPU implementation
  - Runs on same targets
  - Requires SM 3.0 for best performance
- Xeon Phi implementation *in the works*



What we have

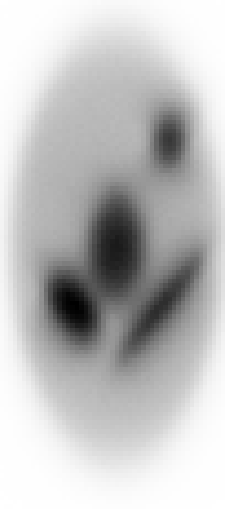
- C++11 generic CPU implementation for 2D x/y and z/y axis
- CUDA GPU accelerated implementation
- Xeon Phi implementation *in the works*
  - Translates C++ into OpenCL SPIR using modified *Clang*
  - Runs on *Linux*
  - *Proof-of-concept*



Example

10^6 events

(a) Phantom



(b) Phantom (detected)

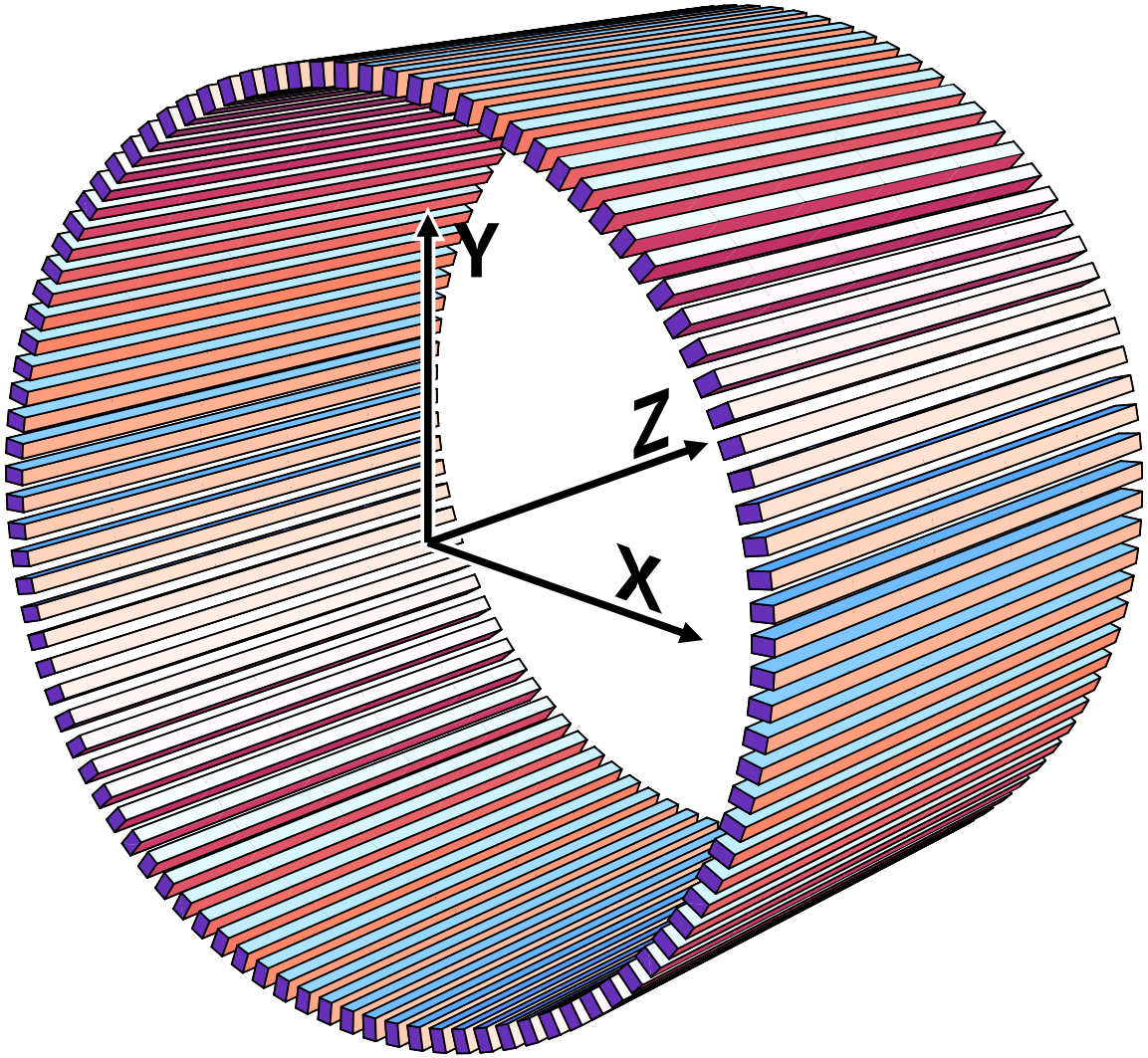
(c) Iteration 1

(d) Iteration 5

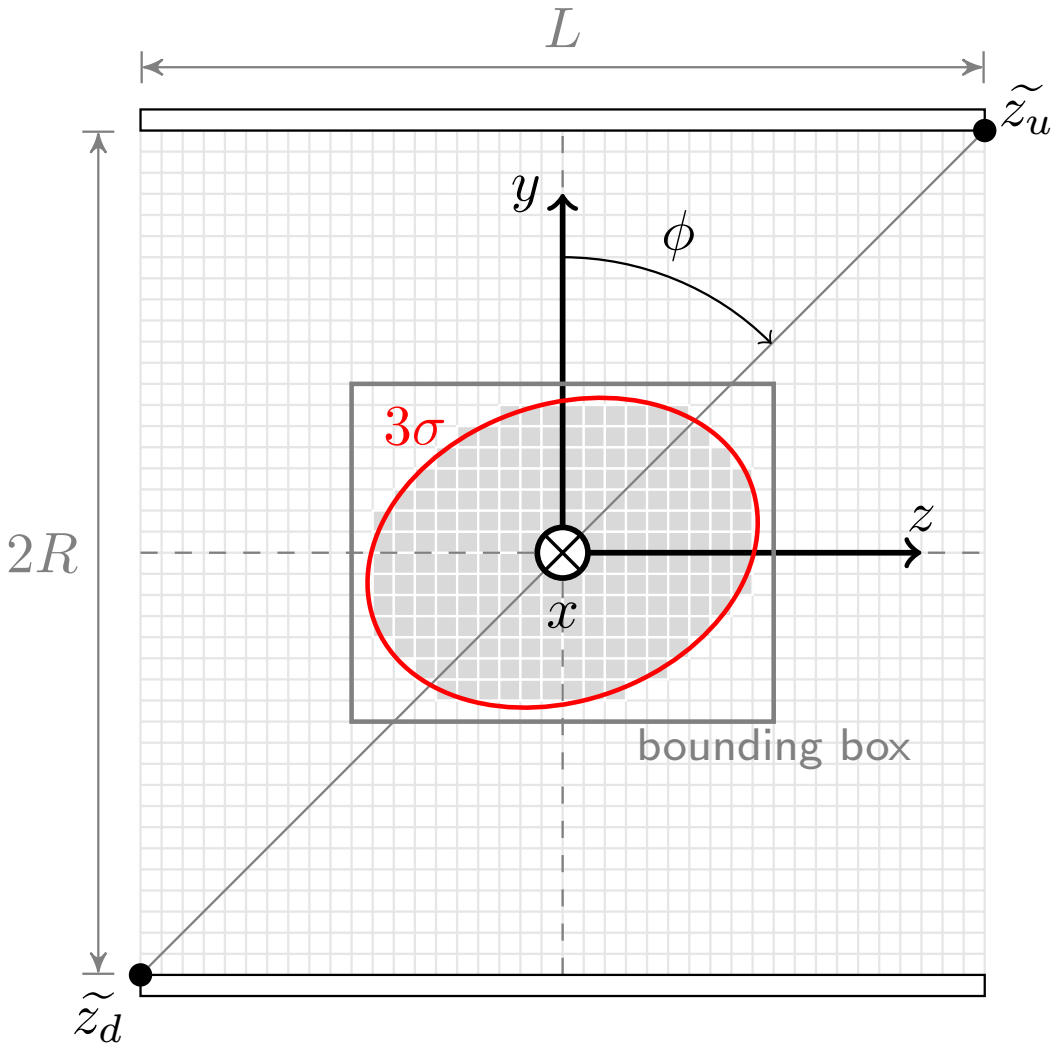
(e) Iteration 10

(f) Iteration 20

3D complete detector geometry



2D frame detector image space



Reconstruction algorithm

$$\rho(l)^{(t+1)} = \sum_{j=1}^{N} \frac{P(\widetilde{\mathbf{e}}_j|l)\rho(l)^{(t)}}{\sum_{i=1}^{M} P(\widetilde{\mathbf{e}}_j|i)s(i)\rho(i)^{(t)}}$$

Naive implementation

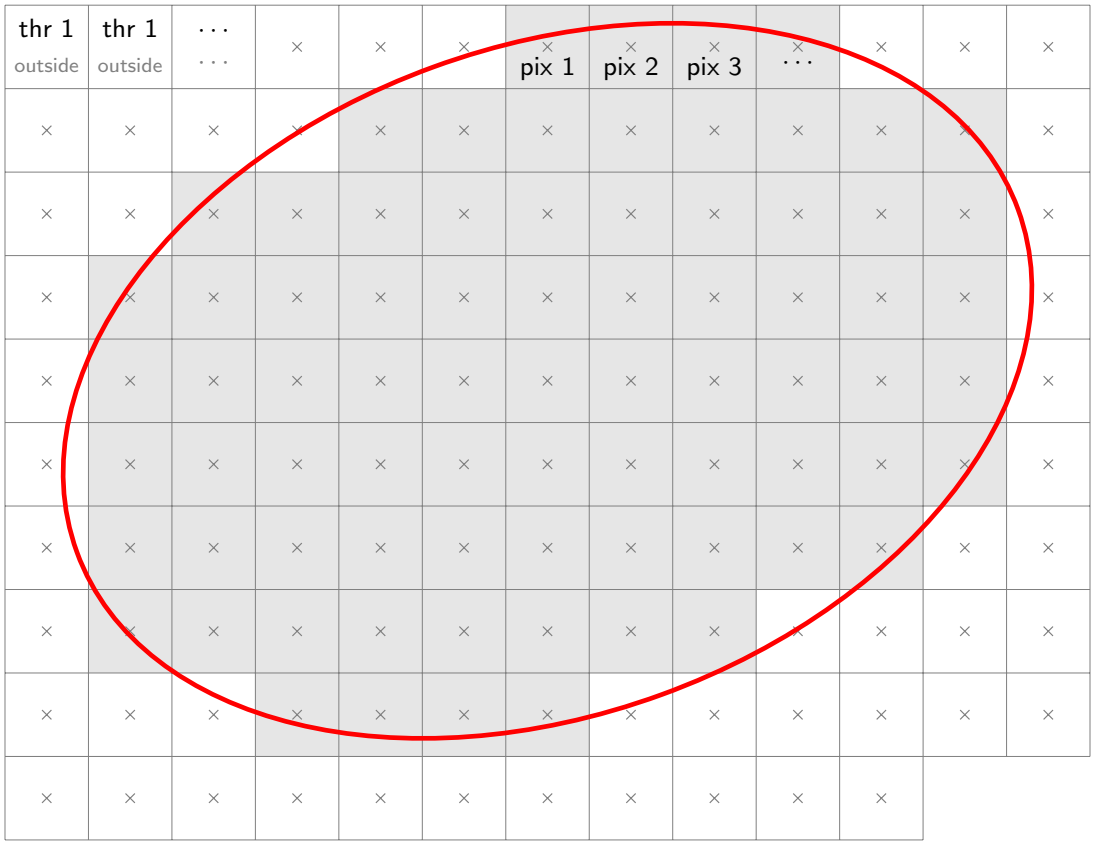
```
for (auto l : pixels) {
  rho_new[l] = 0.0;
  for (auto e_j : events) {
    auto denominator = 0.0;
    for (auto i : pixels) {
      denominator += p(e_j, i) * rho[i];
    }
    rho_new[l] += rho[l] * p(e_j, l)
                  / (denominator * s(l));
  }
}
rho = rho_new;
```

Reconstruction algorithm

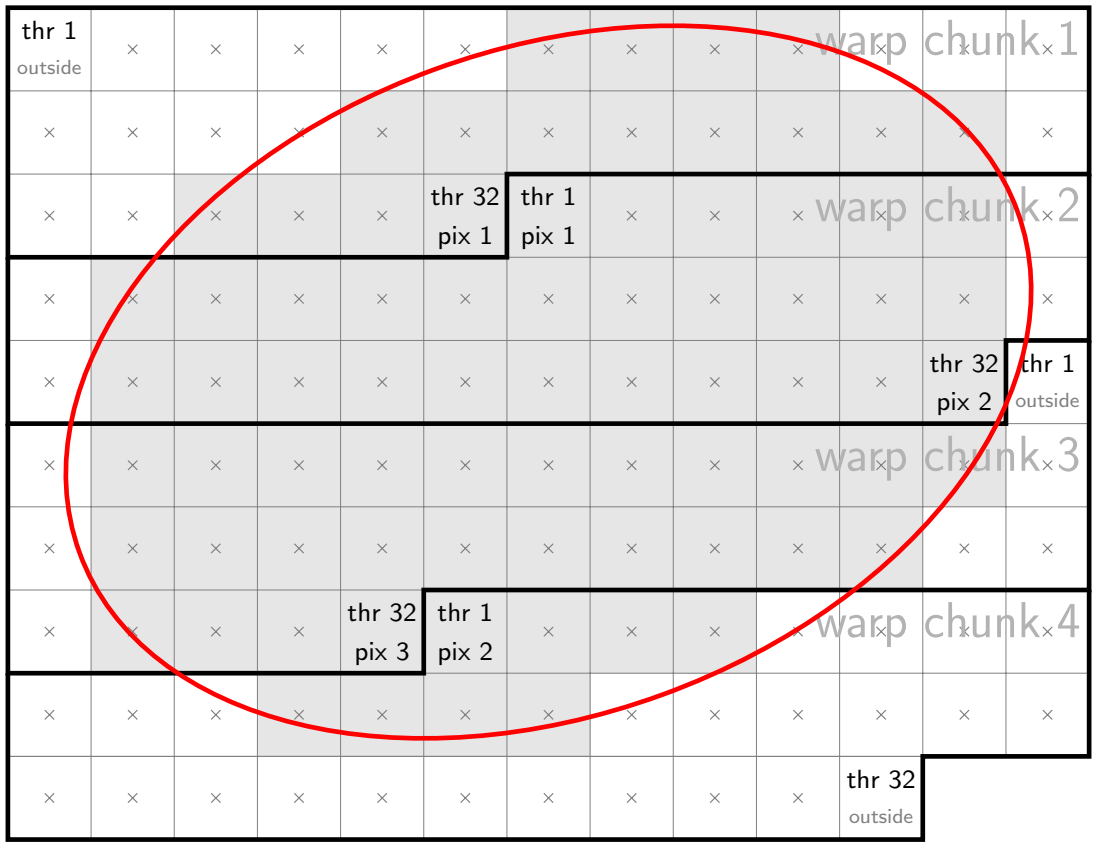
Final implementation

```
for (auto l : pixels) {
  rho_new[l] = 0.0;
}
for (auto e_j : events ) {
    auto denominator = 0.0;
    for (auto i : ellipse(e_j)) {
      kernel[i] = p(e_j, i);
      denominator += kernel[i] * rho[i];
    }
    for (auto i : ellipse(e_j)) {
      rho_new[i] += rho[l] * kernel[i]
                    / (s(i) * denominator);
  }
}
rho = rho_new;
```

Thread granularity (whole event processed by single thread)



Warp granularity (whole event processed by single warp)



GPU reconstruction code blocks time

Kernel	Sensitivity	In-Ellipse	Time
■	■	■	580 ms
■	□	■	573 ms
□	□	■	523 ms
■	■	■	740 ms
■	□	■	706 ms
□	□	■	586 ms
■	■	■	640 ms
■	■	□	483 ms
■	■	□	558 ms
■	■	■	770 ms
■	□	■	734 ms
□	□	■	621 ms
□	□	□	524 ms

■ compile time enabled

□ compile time disabled

■ run-time enabled

□ run-time disabled

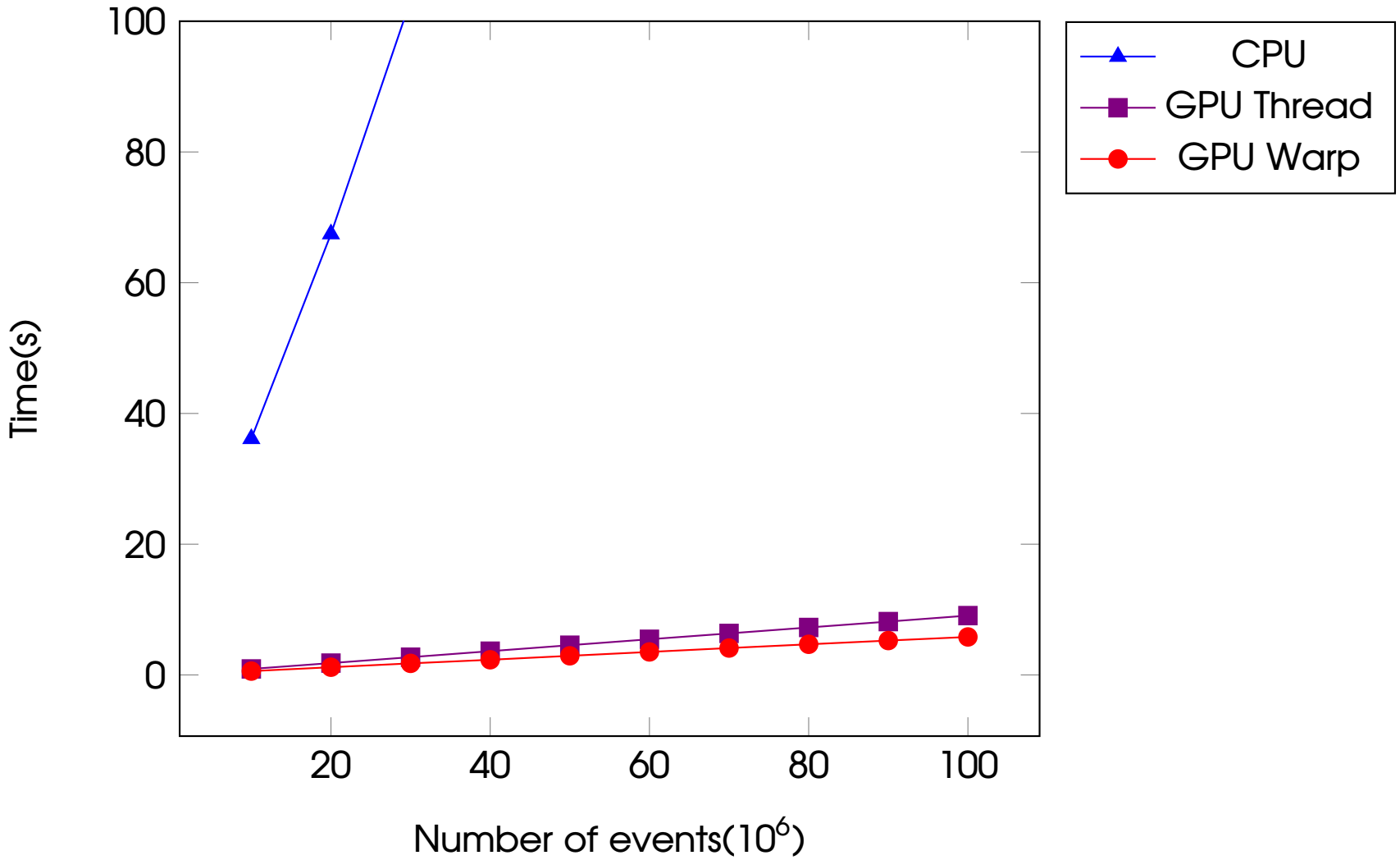
GPU reconstruction time per one iteration

№ of Events	Iteration time
$10 * 10^6$	585 ms
$20 * 10^6$	1 180 ms
$30 * 10^6$	1 770 ms
$40 * 10^6$	2 321 ms
$50 * 10^6$	2 926 ms
$60 * 10^6$	3 519 ms
$70 * 10^6$	4 115 ms
$80 * 10^6$	4 688 ms
$90 * 10^6$	5 258 ms
$100 * 10^6$	5 811 ms

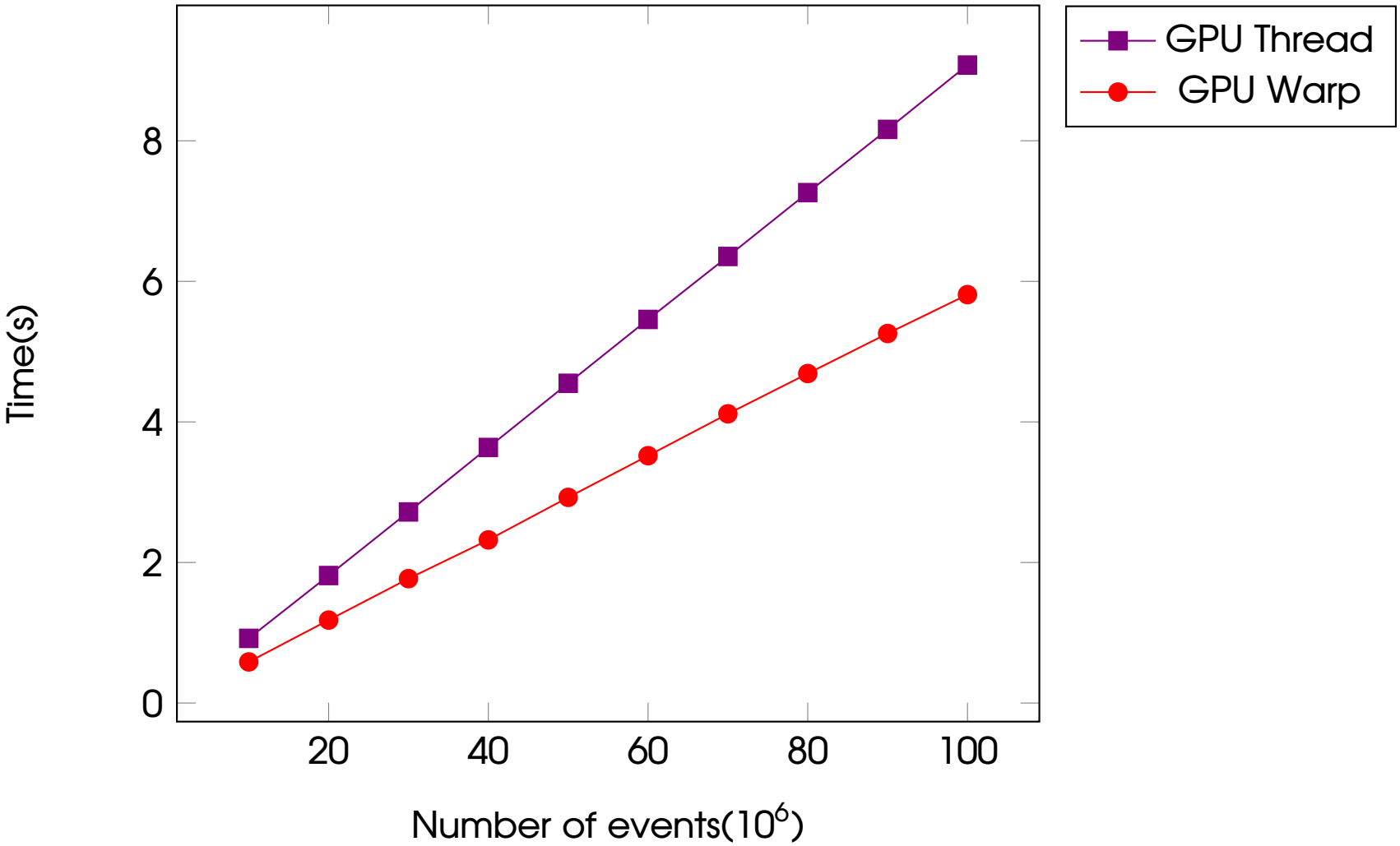
CPU reconstruction time per one iteration

№ of Events	Iteration time
$10 * 10^6$	36 128 ms
$20 * 10^6$	67 448 ms
$30 * 10^6$	103 176 ms
$40 * 10^6$	142 880 ms
$50 * 10^6$	176 328 ms
$60 * 10^6$	209 656 ms
$70 * 10^6$	258 592 ms
$80 * 10^6$	293 496 ms
$90 * 10^6$	313 120 ms
$100 * 10^6$	360 408 ms

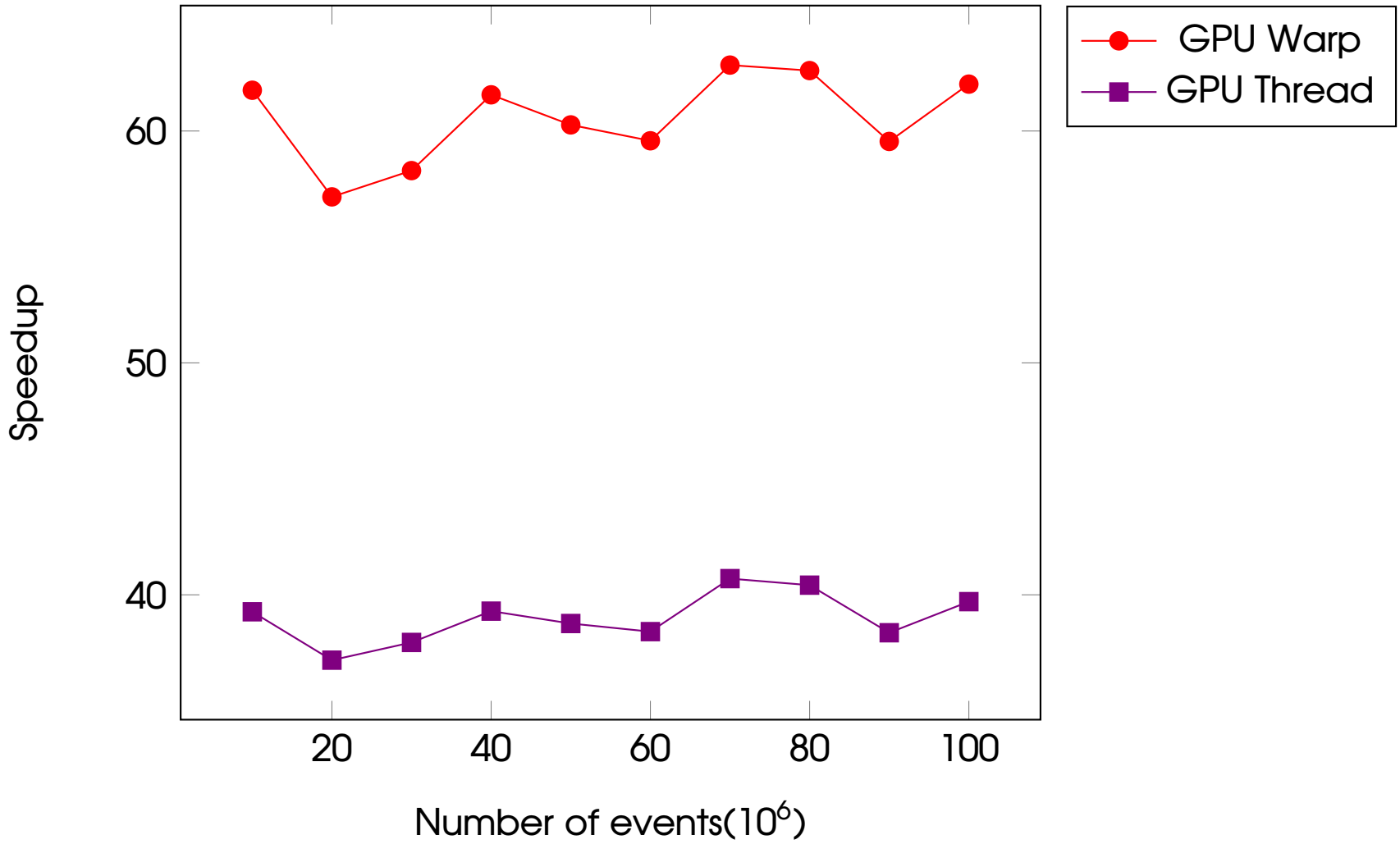
GPU vs CPU implementation (time depending on number of events)



GPU thread vs warp granularity



GPU implementations speedup to CPU implementation



Thank you!

Questions?