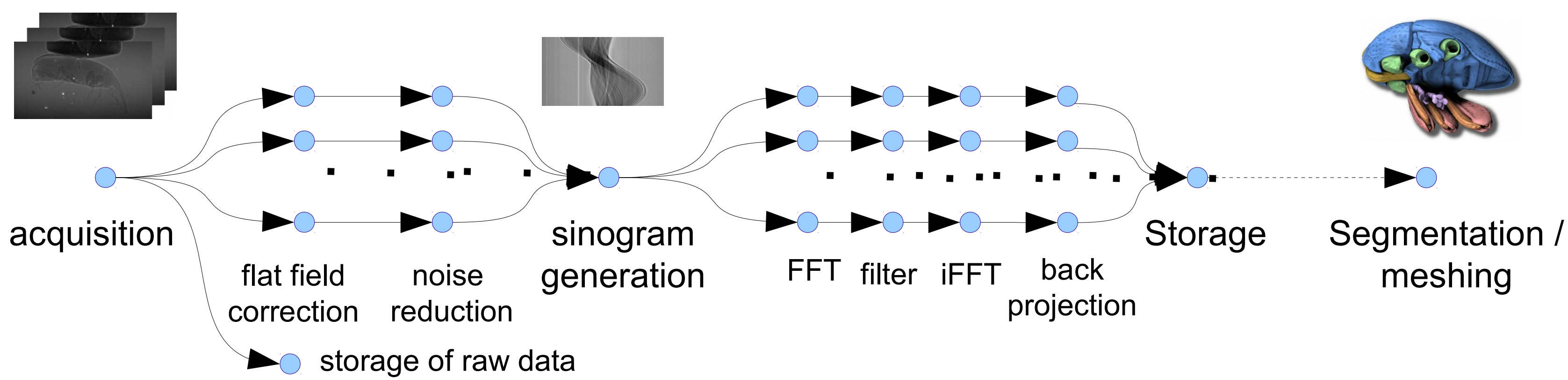
Performance-oriented instrumentation for high-speed synchrotron imaging

S. Chilingaryan, M. Caselle, T. Dritschler, A. Kopmann, A. Shkarin, M. Vogelgesang

- Institute for Data Processing and Electronics at KIT
- Instrumentation for high-speed synchrotron imaging
- Optimizing tomographic reconstruction for parallel architectures



Karlsruhe Institute of Technology

KIT is the merger of Karlsruhe Research Center and *Karlsruhe University*

Research Center

Karlsruhe University

ANKA Synchrotron

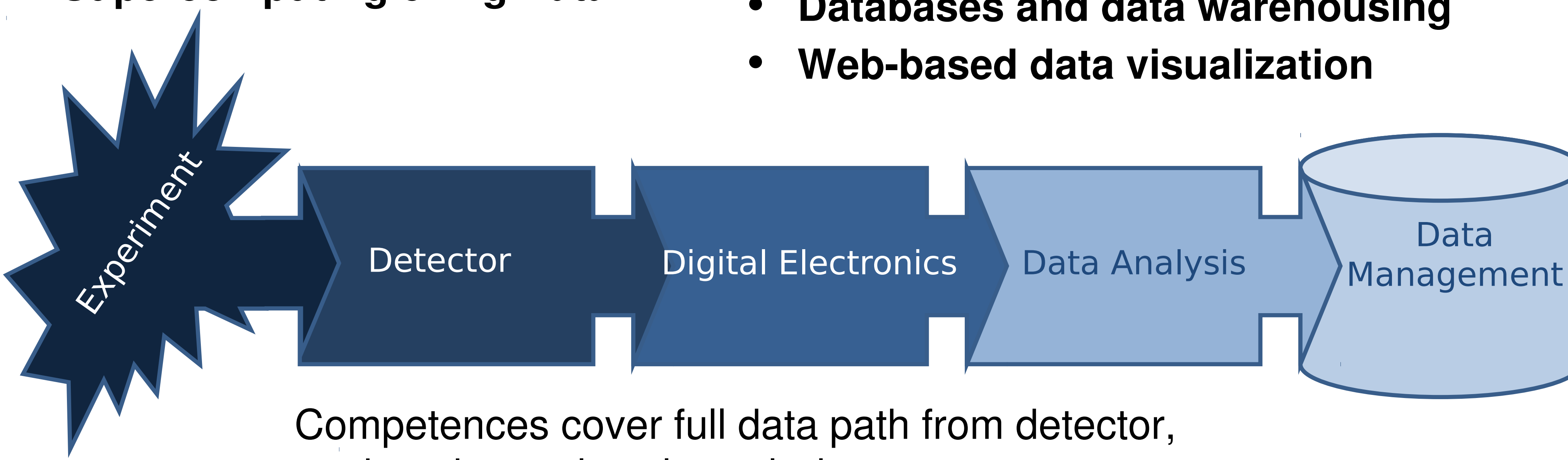
IPE Competences

Experiments

- Astroparticle & High Energy Physics
- Atmosphere and Climate
- Nuclear Fusion
- Electrical Storage Systems
- Photon Science
- Ultrasound Tomography
- Nano- and Microsystems
- Supercomputing & Big Data

Tools and Technologies

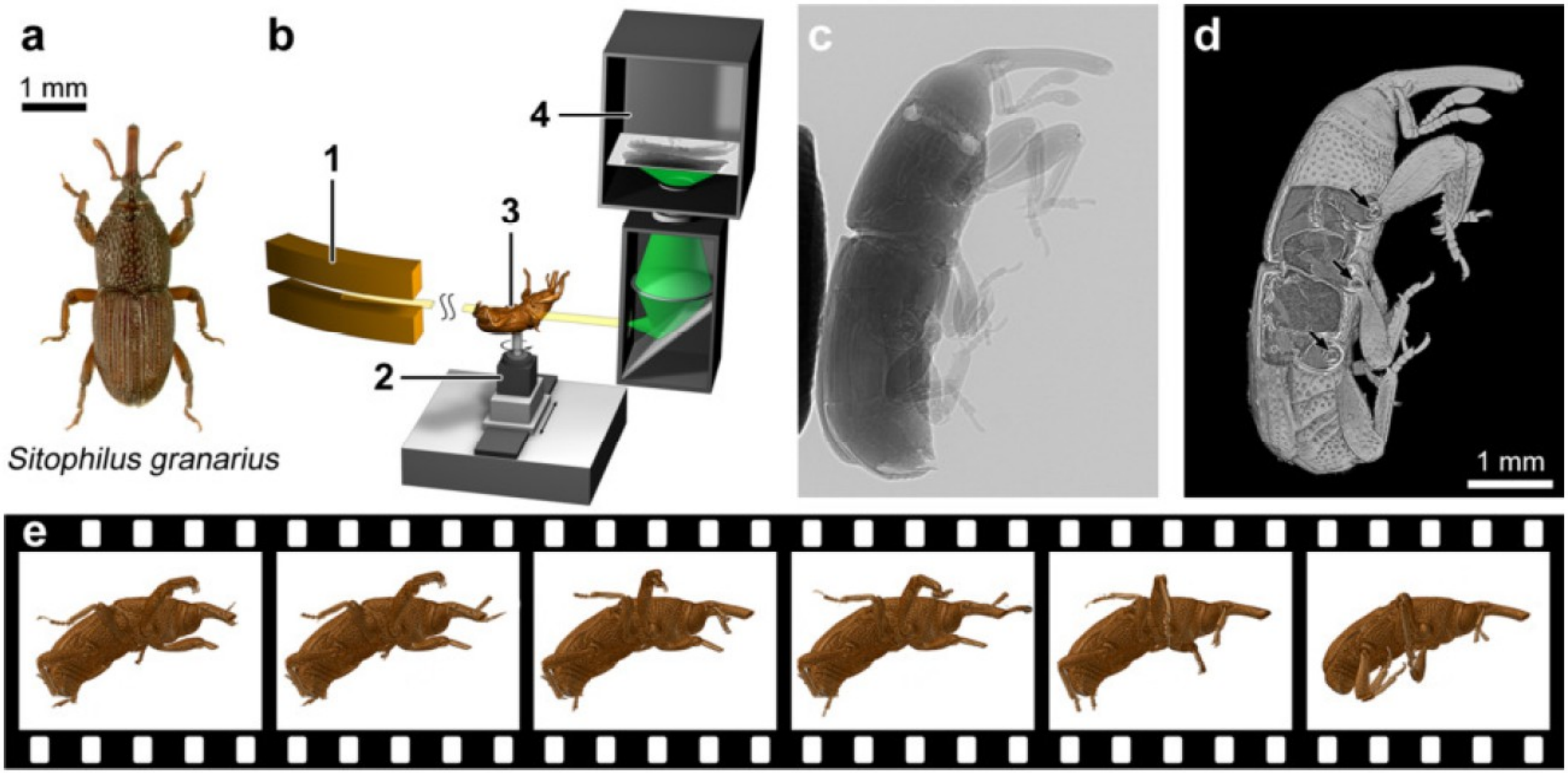
- High-speed DAQ Electronics
- High-performance and GPU computing
- Software optimization
- Databases and data warehousing
- Web-based data visualization



Competences cover full data path from detector, analog electronics, through data management

Prof. Dr. Marc Weber
Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Example 4D cine-tomography experiment



In vivo X-ray 4D cine-tomography experiment. (a) Photograph of Sitophilus granarius, dorsal view. (b) Experimental set-up for ultra-fast X-ray microtomography showing bending magnet (1), rotation stage (2), fixed specimen (3) and detector system (4). (c) Radiographic projection. (d) 3D rendering of the reconstructed volume with thorax cut open and revealing hip joints (arrows). (e) In vivo cine-tomographic sequence of moving weevil.

Scope of the problem & Projects

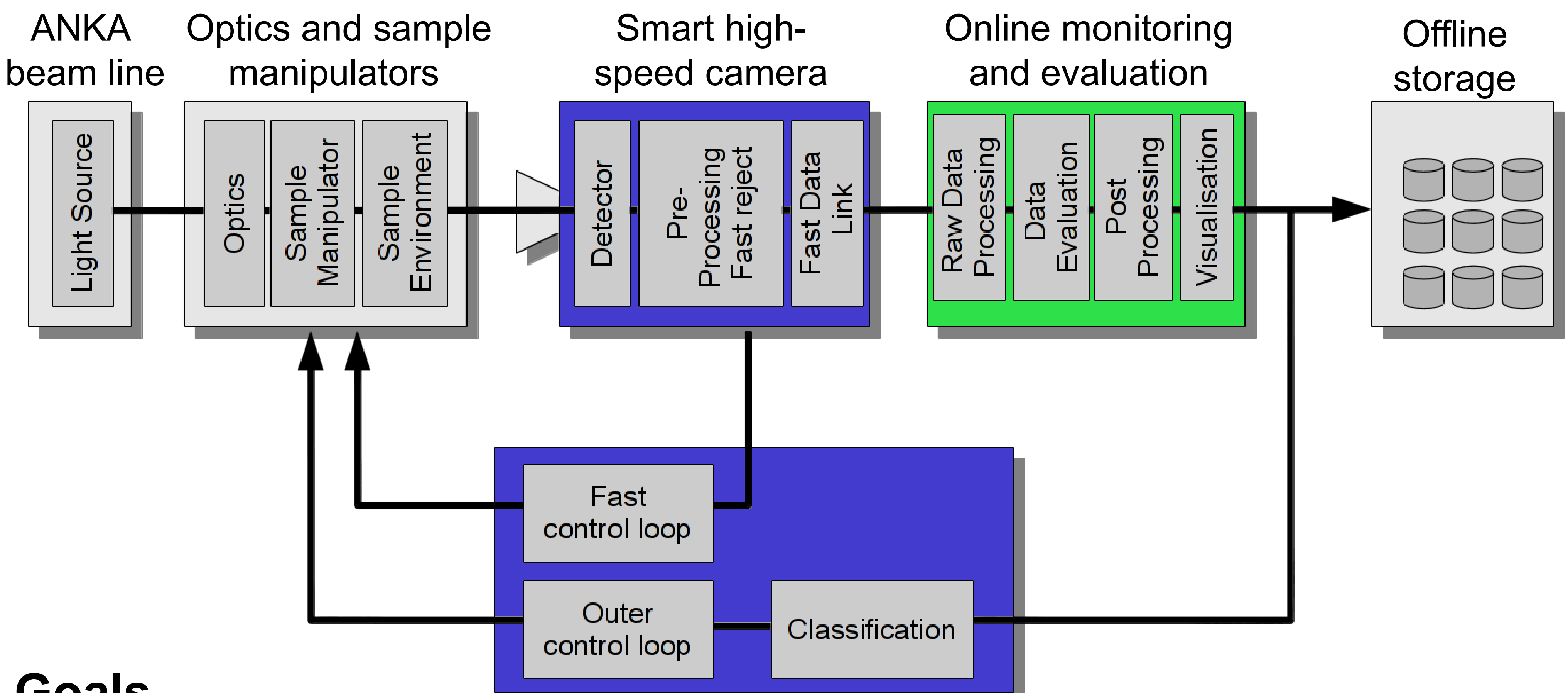
Fully automated 4D imaging of living species with high spatial and temporal resolution and image-based control

UFO	STROBOS	ASTOR
Online montiroing and image-based control	**Low Dose Laminography**	**Post-processing tools for biologists**
Real-time reconstruction and Visualization	High-quality reconstruction from under-sampled data for diffraction laminography	Work-flow for remote semi-automated segmentation

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

UFO *Ultra Fast X-ray Imaging of Scientific Processes with On-Line Assessment and Data-Driven Process Control*



Goals

- High speed tomography
- Increase sample throughput
- Tomography of temporal processes
- Allow interactive quality assessment
- Enable data driven control
  - Auto-tunning optical system
  - Tracking dynamic processes
  - Finding area of interest

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

4D Tomography of living organisms

Radiation

Motion

Radiation is destructive and limits duration of experiment

Motion during the acquisition of projections is blurring the reconstruction

We need to reduce number of used projections

Nyquist-Shanon criteria defines a minimum number of projections required for quality reconstruction

A priori knowledge can be introduced to overcome the restriction

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Reconstruction Algorithms

Karlsruhe Institute of Technology

Analytic

DFM

DFI
Gridding
Pasciak

FBP

Faster

More Robust

High performance

Iterative

ART

SIRT
SART
OS-SART

Minimization techniques

Shrinkage
SD
CGLS
...

Compound methods

ASD-POCS
Split-Bregman
...

+ Geometry Modeling
+ Projection Modeling
+ A priory Knowledge

Customizable Reconstruction

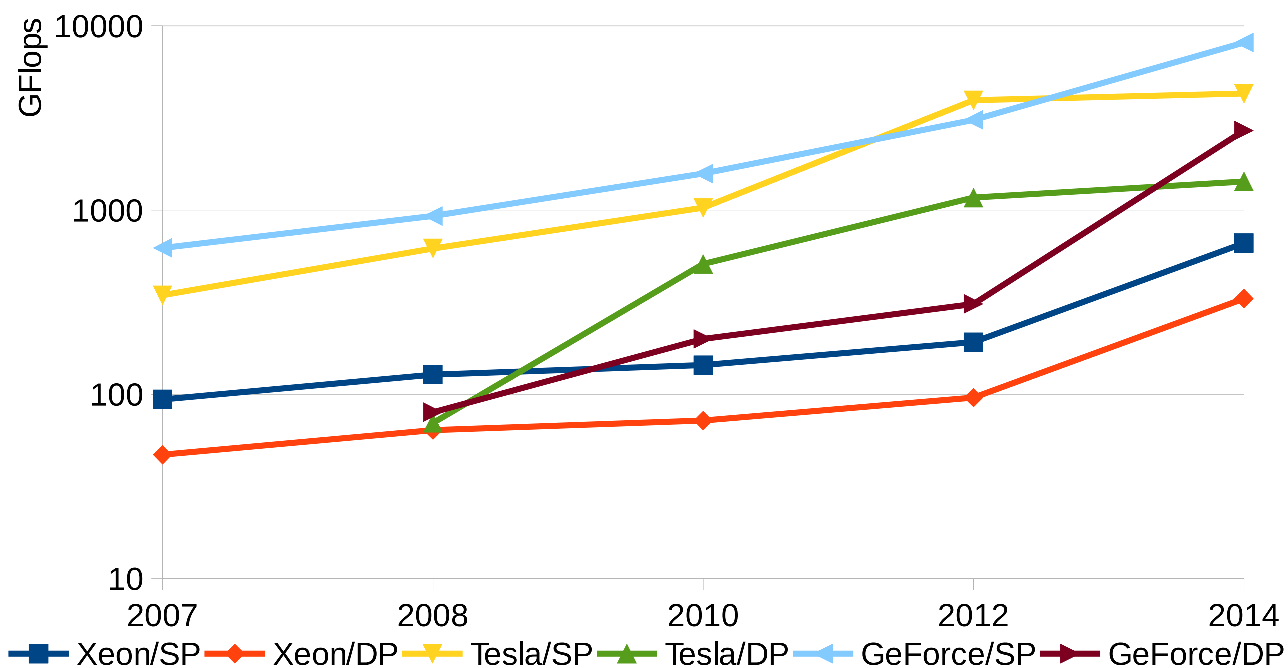
S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Handling the computational problem

- Distributed control system based on Infiniband interconnects
- GPU-based computing
- Multiple levels of scalability
- Cheap off-the-shelf components
- Modular reconstruction framework

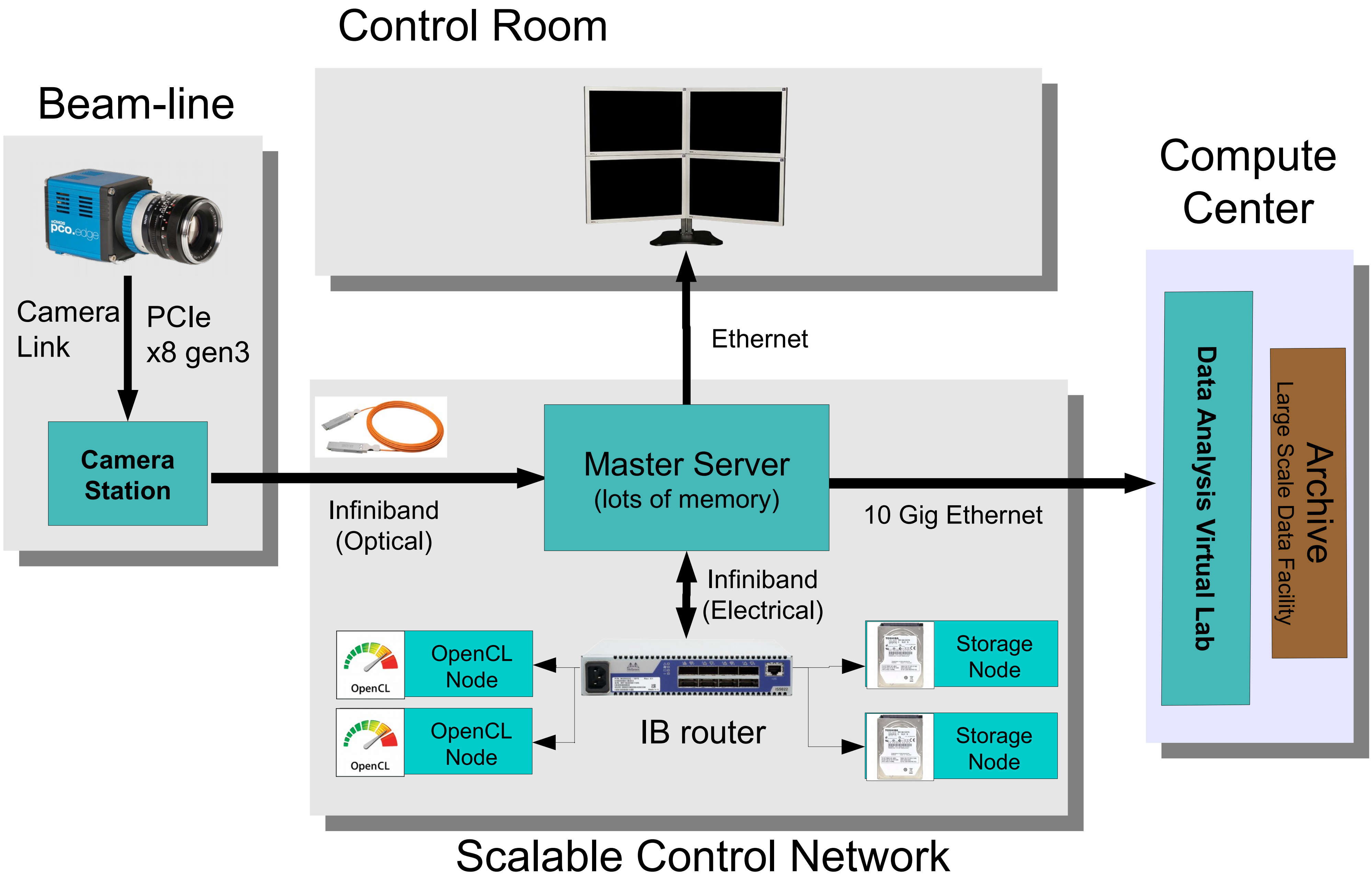
Easily scalable
Up to 4 GPUs
with 35 Tflops
for ~ 5000 EUR



Historical trends of CPU and GPU performance

S. Chilingaryan et. all
Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

UFO Control Network



S. Chilingaryan et. all

Master Server

Cameras

Internal
PCO.edge
PCO.dimax
….

FDR Infiniband
56 Gbit/s

Ethernet
10 Gb/s

Storage

LSDF
Large Scale Data Facility

External PCIe x16 (16 GB/s)

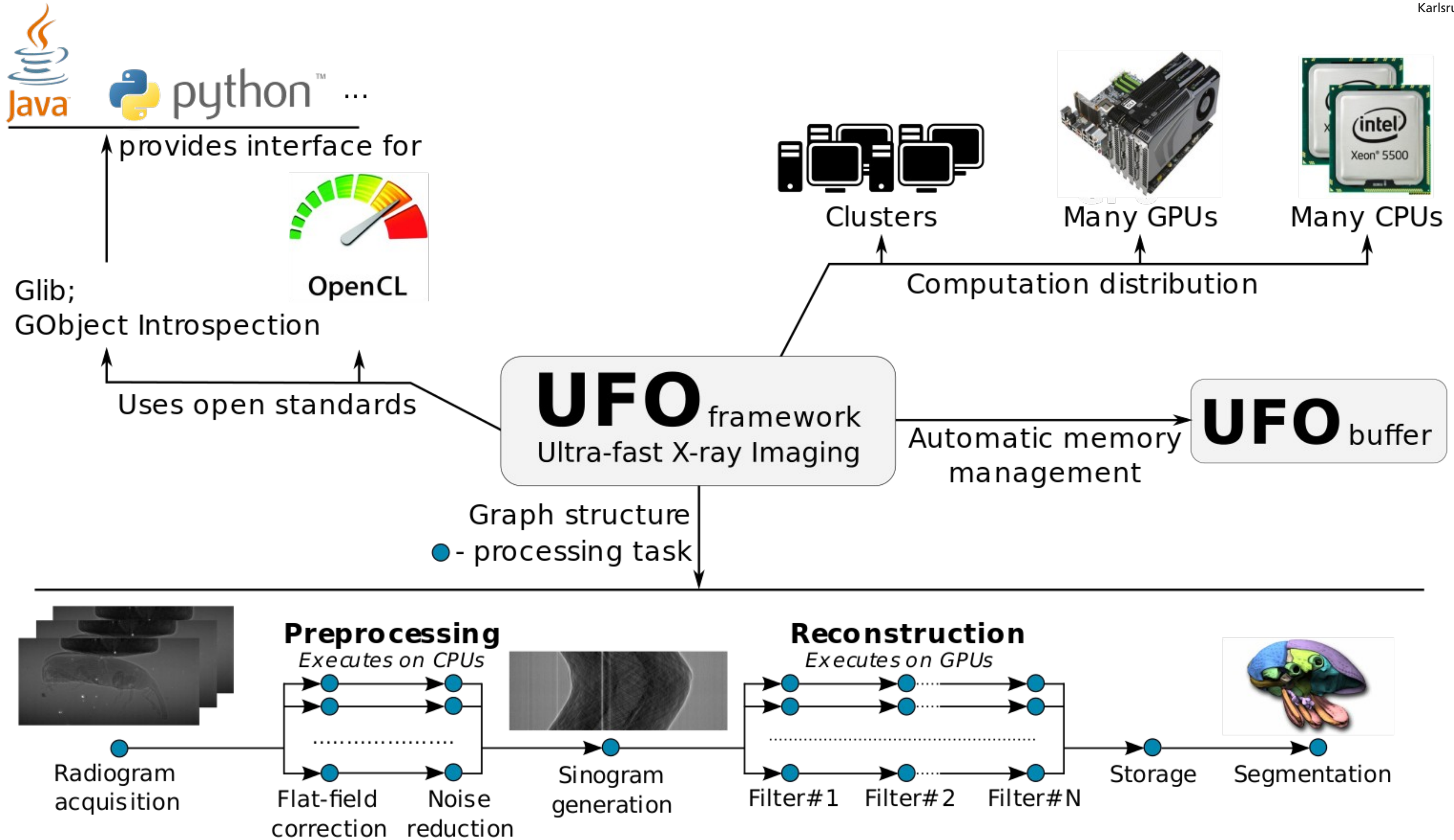
SFF8088 (2.4 GB/s)

<u>SuperMicro 7047GR-TRF</u> (Intel C602 Chipset)
CPU: 2 x Xeon E5-2680v2 (total 20 cores at 2.8 Ghz)
GPUs: 7 x NVIDIA GTX Titan
Memory: 256 GB (512GB max)
Network: Intel 82598EB (10 Gb/s)
Infiniband: 2 x Mellanox ConnectX-3 VPI
Storage: Areca ARC-1880-ix-12 SAS Raid
8 x Samsung 840 Pro 510 (Raid0)
16 x Hitachi A7K200 (Raid6)

- High amount of memory
- Fast SSD-based Raid for overflow data
- Easy scalability with external PCI express and SAS

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

UFO Image Processing Framework



Fully pipelined architecture supporting diversity of the hardware platforms and based on open standards for easy algorithms exchange. Easy prototyping with Python and other scripting languages.

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

UFO Filters

Preprocessing

Flat-field correction
Phase-contrast Imaging
Grating Interferometry

Tomography

Filtered Back Projection
Direct Fourier Inversion
SART / SIRT

Laminography

Filtered Back Projection
Discrete ART

Projectors

Joseph
DFI-based

Regularized

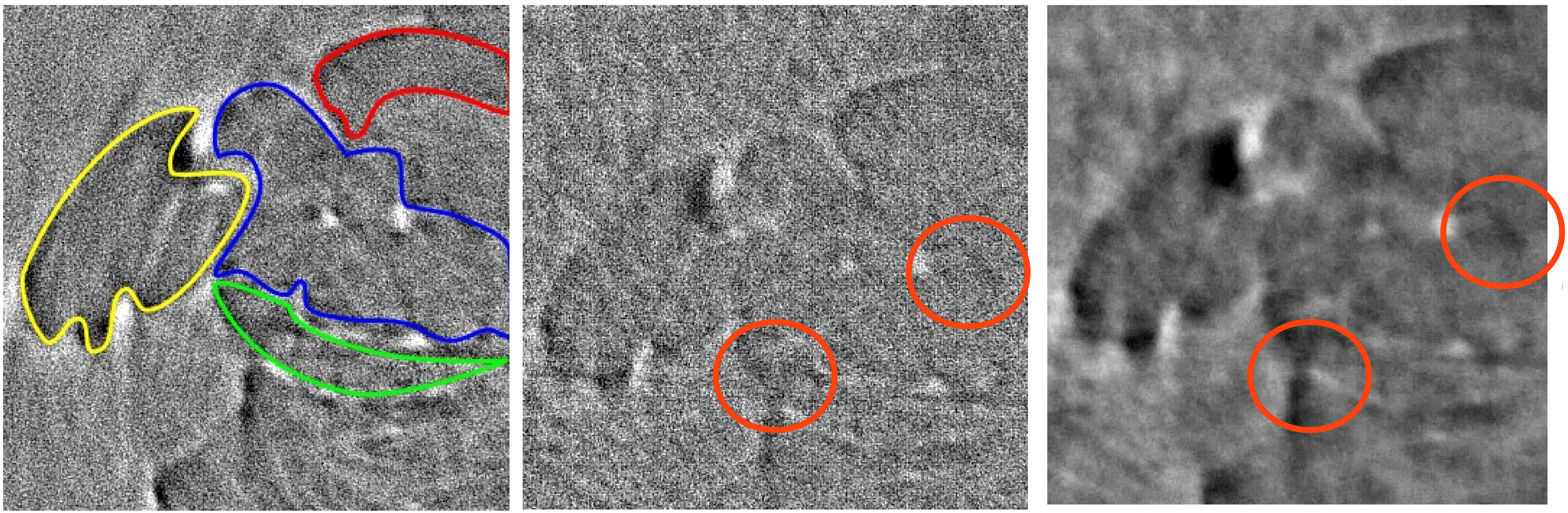
SBTV
ASD-POCS
Split-Bregman / TV
Split-Bregman / *lets
Split-Bregman / Hybrid

Postprocessing

De-noising
Optical flow
Visualization

UFO Algorithms: Synchrotron data

Zoomed joint of Sitophilus granarius (grain weevil)



Segmented | FBP | Split-Bregman with Framelets

Reconstructed from 50 projections (~ 1/40 of standard dataset)

S. Chilingaryan et. all

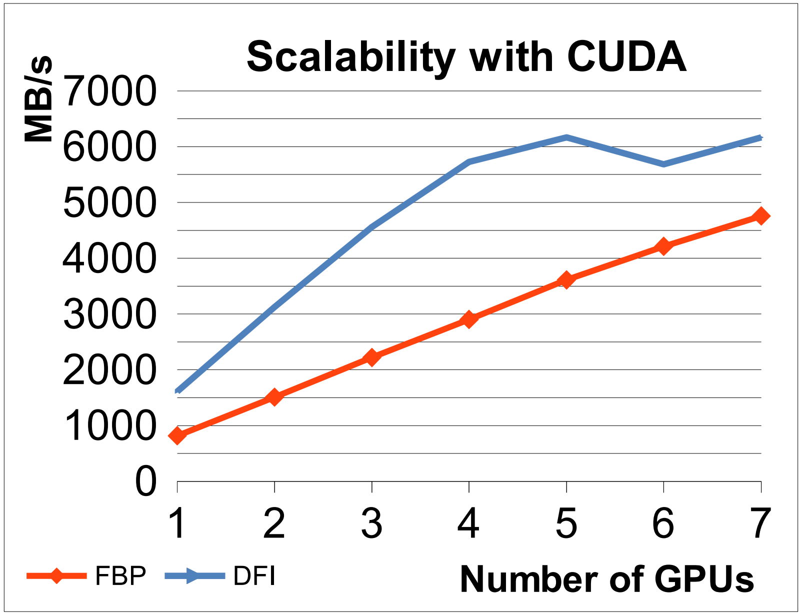
Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Reconstruction Performance

DFI	FBP	SART/SIRT
$N^2 * \log N$	$M * N^2$	N^2 equations
1600 MB/s (CUDA) 850 MB/s (OpenCL)	800 MB/s	500 KB/s

CUDA Scaling up to 6000 MB/s
~ 2250 MB / s from 12 bit camera

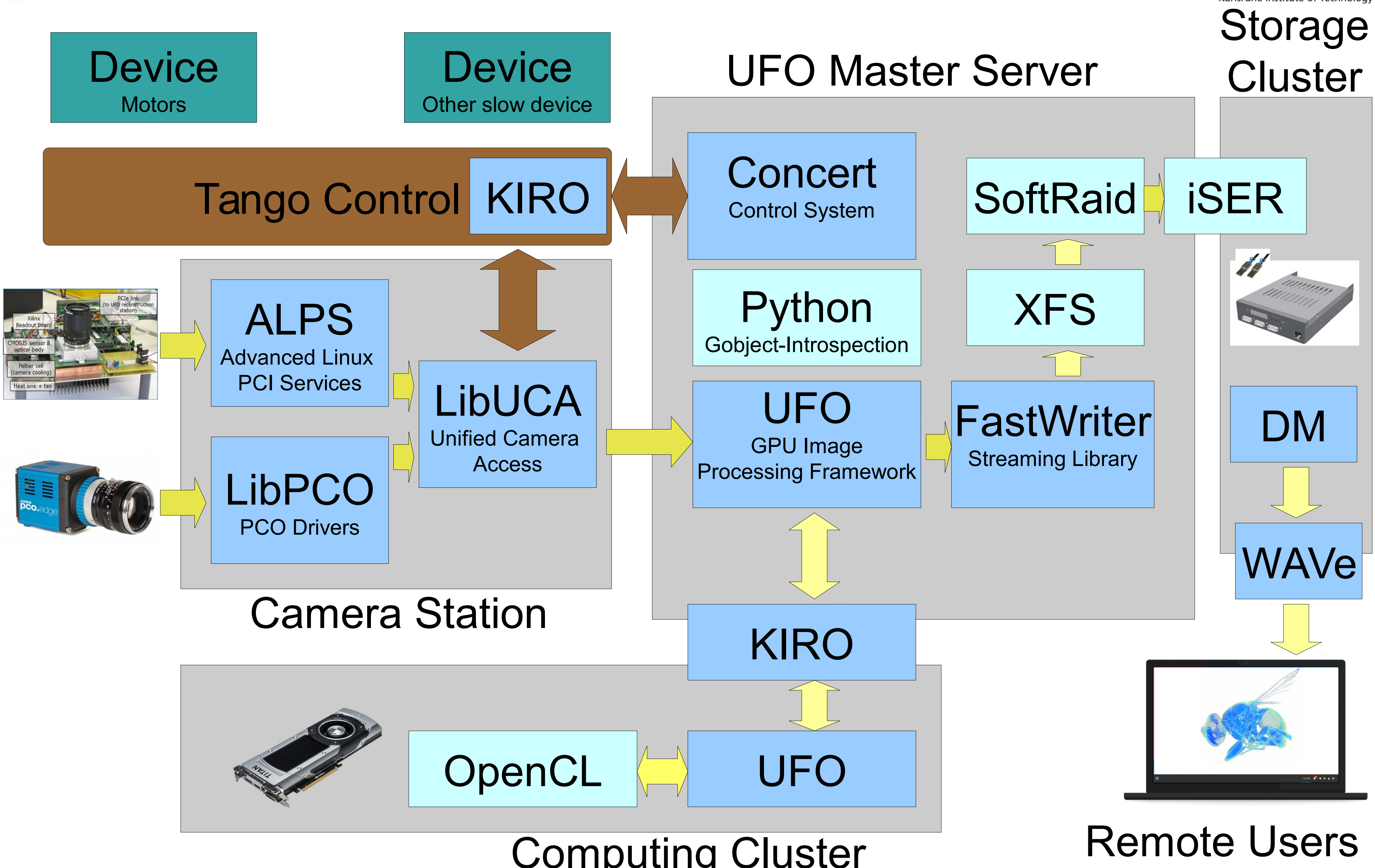
OpenCL Scaling up to 2500 MB/s
~ 950 MB / s from 12 bit camera



S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

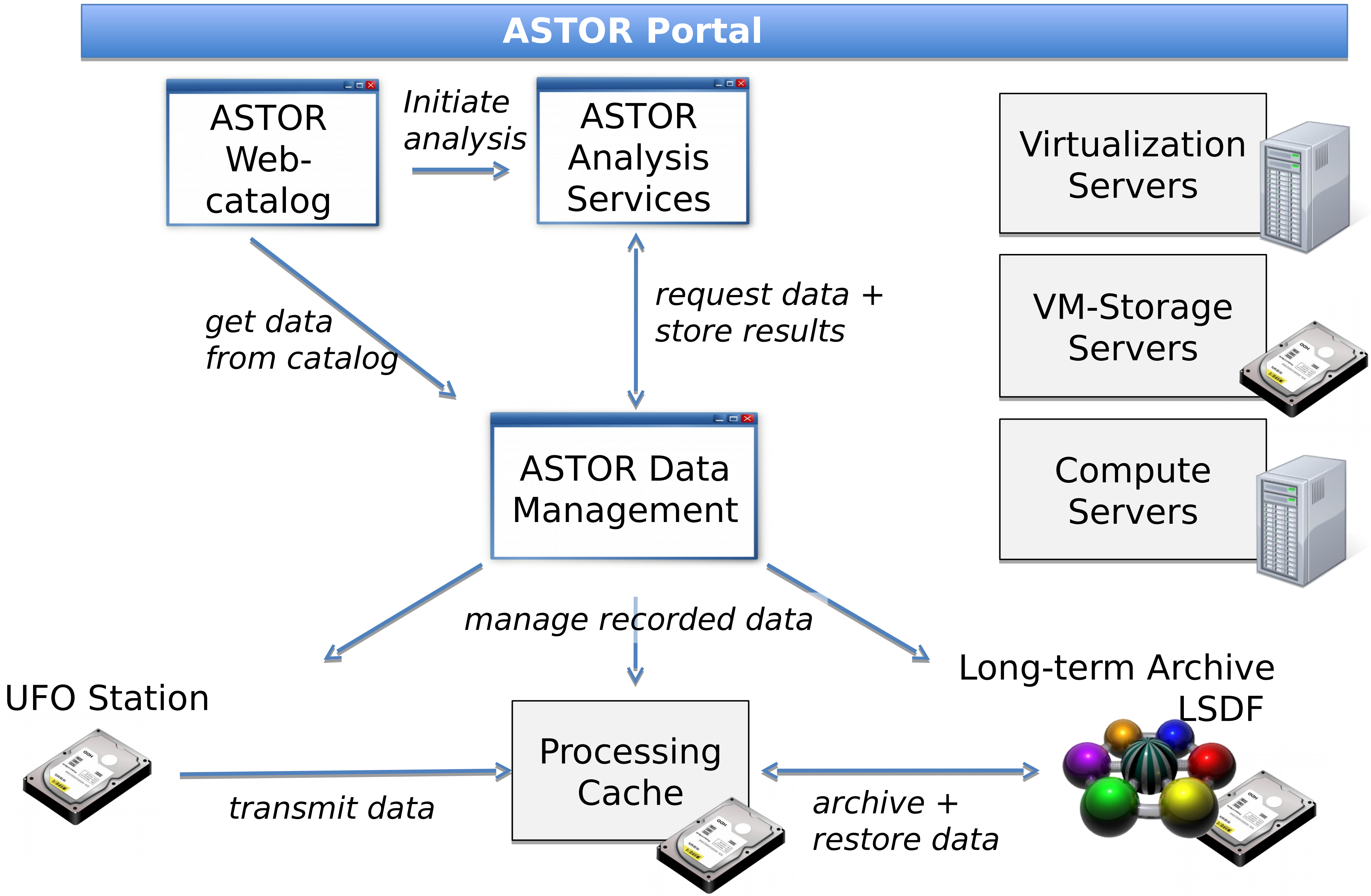
Software Stack



S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

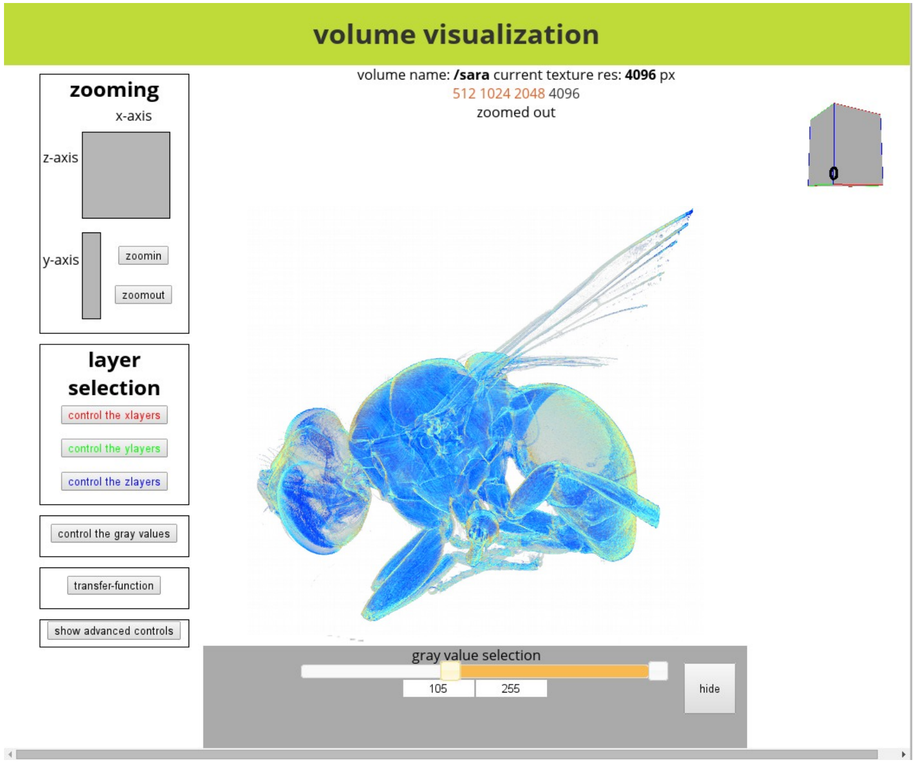
ASTOR



S. Chilingaryan et. all

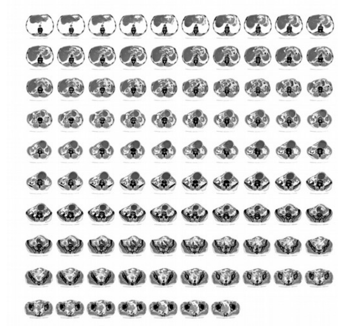
Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

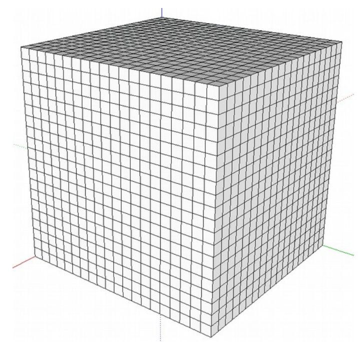
WAVe: Web-based volume visualization



Ray-casting approach

A preview slice-maps pre-generated using UFO framework





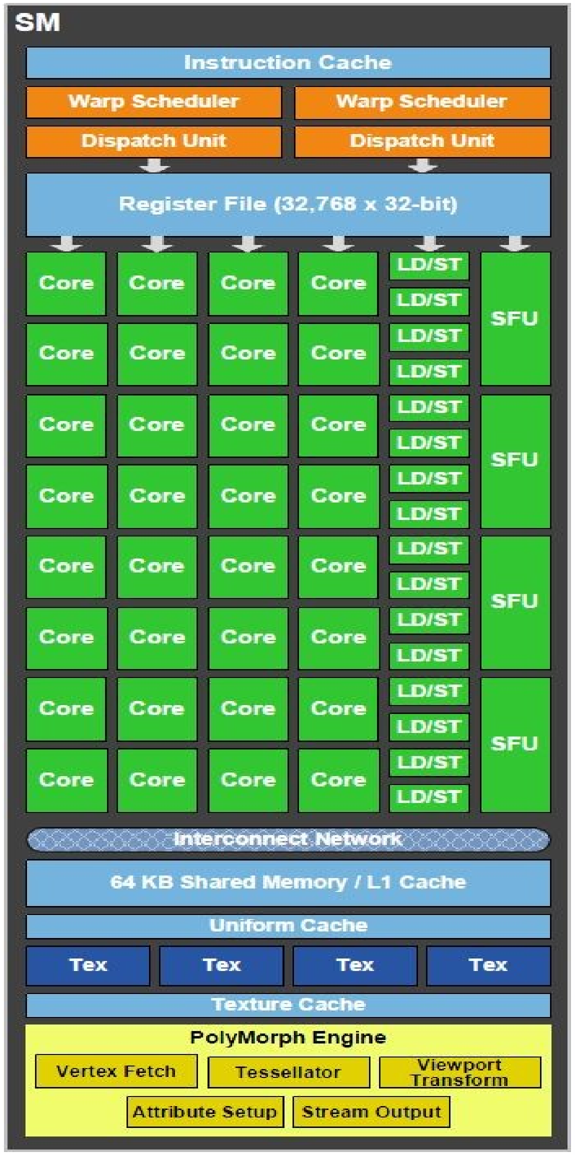
Optimized storage layout for fast zooming

- Working on majority mobile platforms with descent GPUs
- Multiple zooming levels for inspecting fine details
- High-quality cuts
- Automatic thresholding-based segmentation
- Multi-modality rendering support

Optimizing tomography for parallel architectures

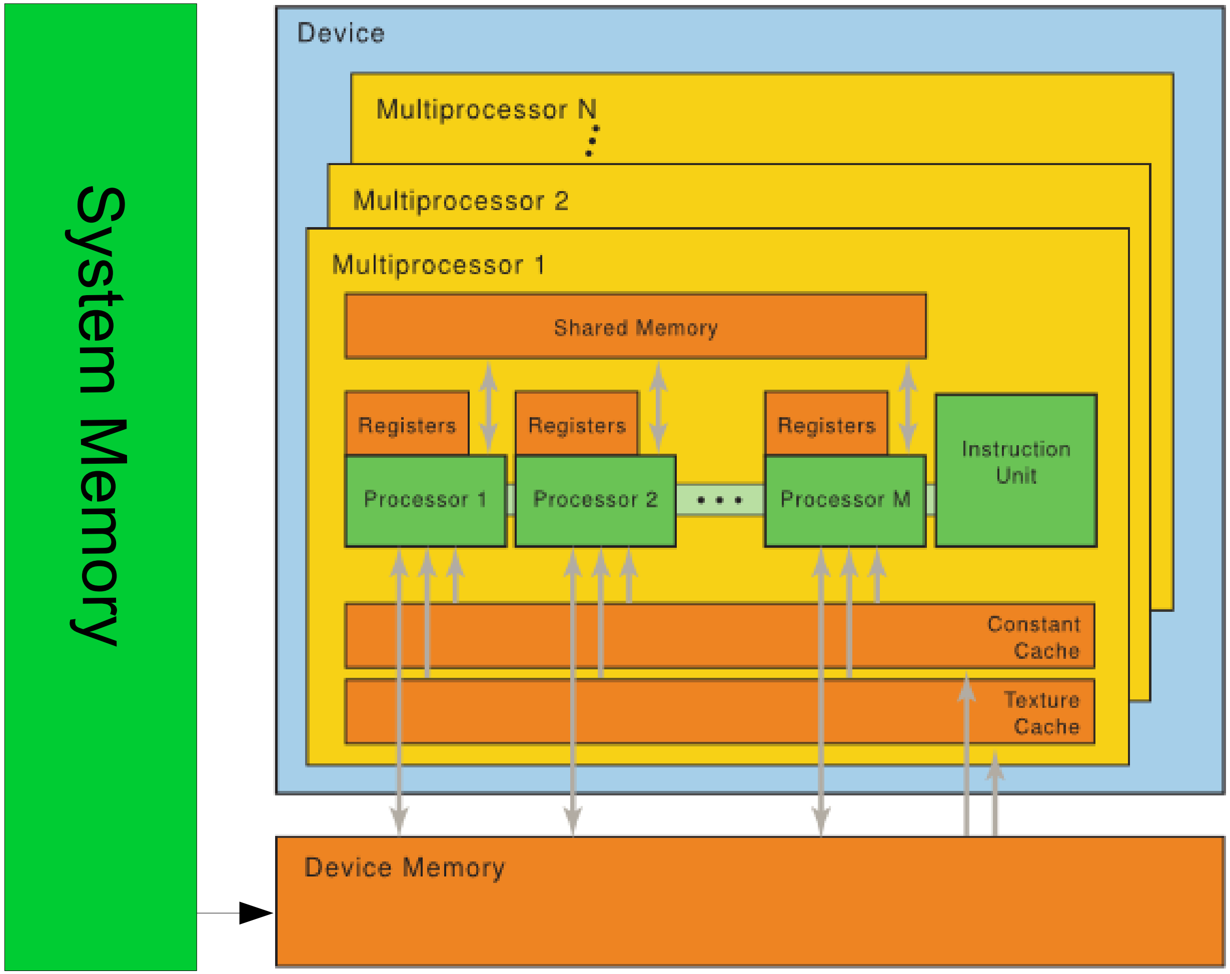
- **Consists of SIMD-type Compute Units (CU)**
  - One instruction is executed on many data items
  - Each CU able to execute several operation types
  - But only FP additions/multiplications are fast
- **Posses complex memory hierarchy**
  - Low Bandwidth-per-flop ratio and small caches
  - Up to four different types of memory
  - Optimal access pattern have to be followed
- **Architectures vary drastically**
  - Sizes, speed, and structure of memories / caches
  - Types and amount of provided processing units
  - Balance of operation throughput

Codes and algorithms have to be carefully optimized for the specific parallel architecture



Compute Unit on Fermi

Memory model



- **Host Memory**
  - 6 GB/s (PCIe x16 gen2) to 12 GB/s (PCIe x16 gen3)
- **Global Memory**
  - 100 – 300 GB/s with latencies up to 1000 clocks
- **Local Memory**
  - 1 – 2 TB/s (total) with latencies below 100 clocks
- **Registers**
  - private to threads
- **Caches**
  - L1/L2 cache
  - Texture cache
  - Constant memory

Complex memory hierarchy consisting of 4 levels and with each level one order of magnitude faster when previous!

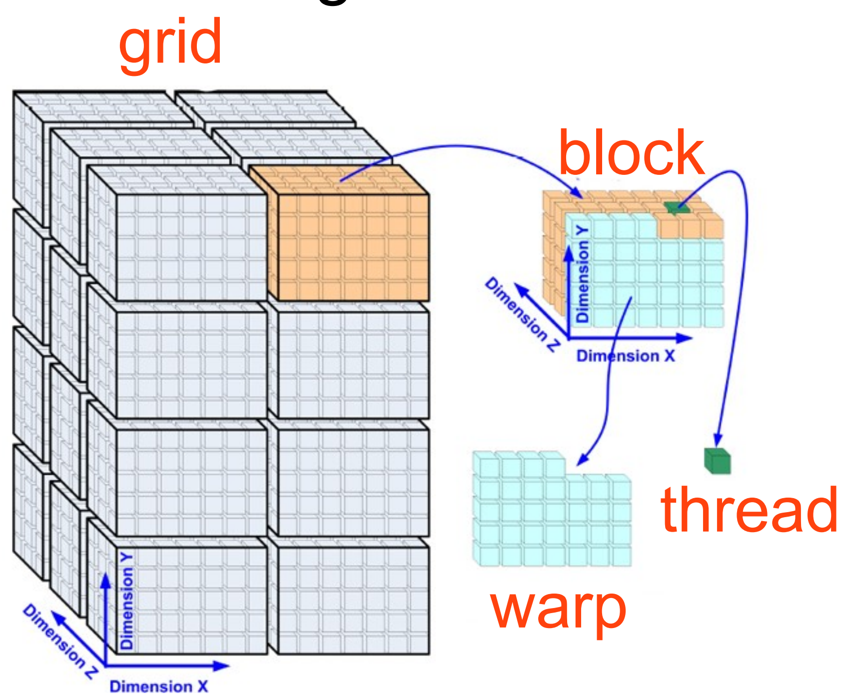
Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Programming Model

Thread abstraction is used to split the problem space into the independent GPU tasks

- All threads execute the same code (**kernel**)
- Task is defined by the linear or volumetric index of the thread
- GPU schedules threads in groups of fixed size (**warp**)
- A user-defined **block** of threads is assigned to a specific CU
- Threads of the block may exchange data using CU shared memory

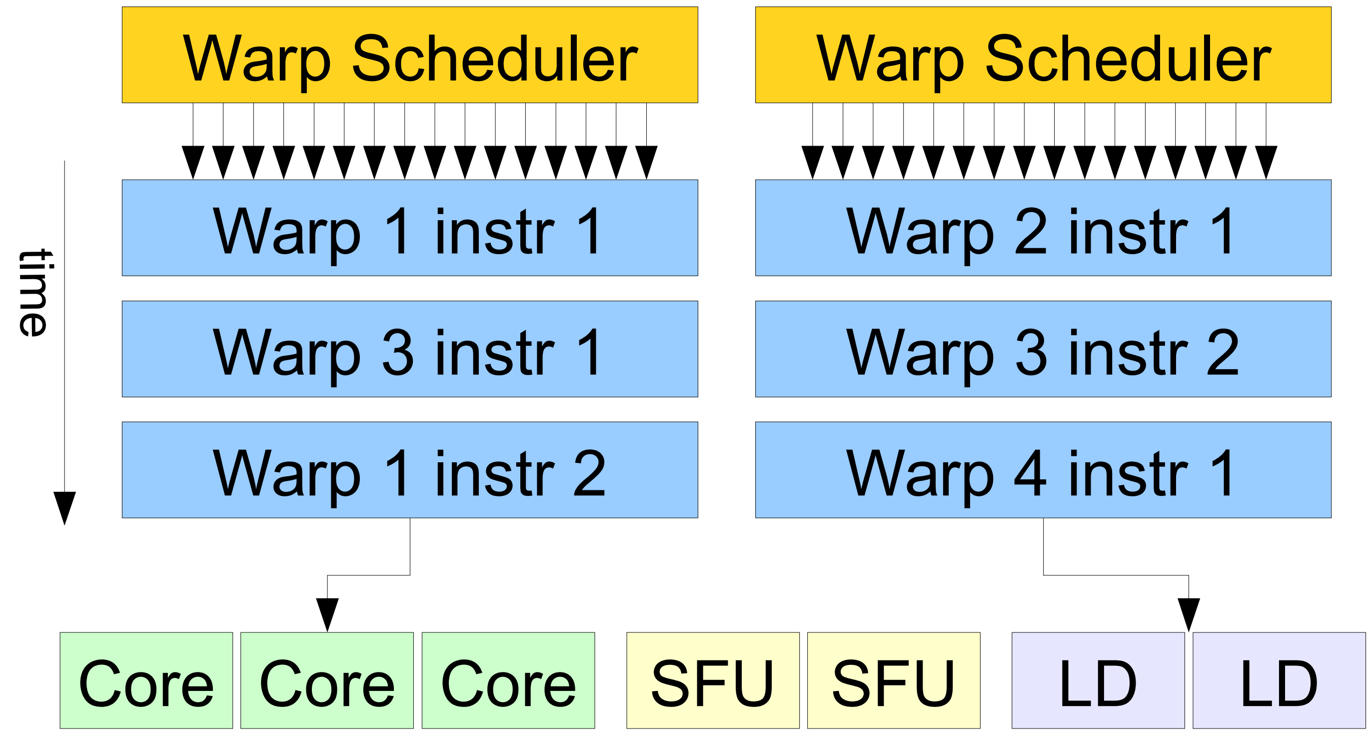
e.g. resulting image is mapped to a 1-, 2-, or 3D grid of GPU threads and each pixel is computed by a thread with the index equal to pixel coordinates



S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Scheduling



Multiple warps on CU executed in parallel

Independent instructions executed in parallel

Warp 4 will be blocked for a long time, but other warps on CU will execute and hide the latency

- Warps from several blocks are executed by CU in parallel
- The number of currently resident warps is called **occupancy**
- Occupancy is limited by available registers and shared memory
- Suboptimal occupancy limits the instruction bandwidth

For optimal performance we have to increase occupancy and number of independent instructions

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

FBP Reconstruction

Karlsruhe Institute of Technology

1. Filtering

Multiplication with the configured filter in the Fourier space

2. BackProjection

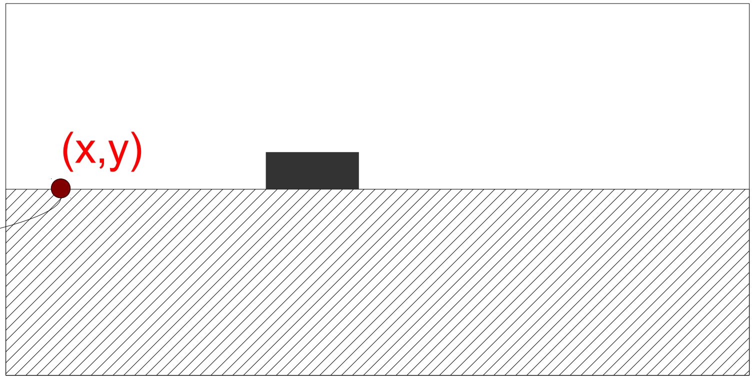
bins →

proj. 0	1	2	3	4	5	….
proj. 1	1	2	3	4	5	….
proj. 2	1	2	3	4	5	….
			….			
proj. α	1	2	3	4	5	….

1. For each position we compute:
   $x \bullet \cos(\alpha) - y \bullet \sin(\alpha)$
2. Interpolate between neighboring bins
3. Sum over all projection
4. The sum is the value of (x,y)

$x \bullet \cos(\alpha) - y \bullet \sin(\alpha)$

For each texel of output volume and for each projection we perform a single linear interpolation



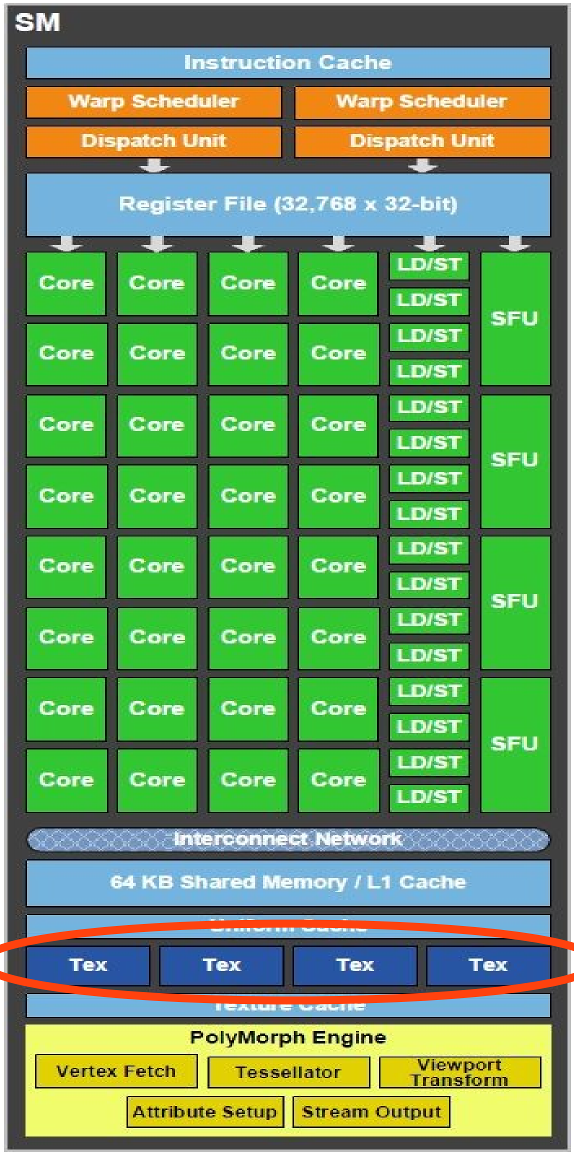
Texture Engine

Features:

- Spatial-aware cache
- Bi/tri-linear interpolation
- Normalized coordinates
- Different clamping modes

Applications:

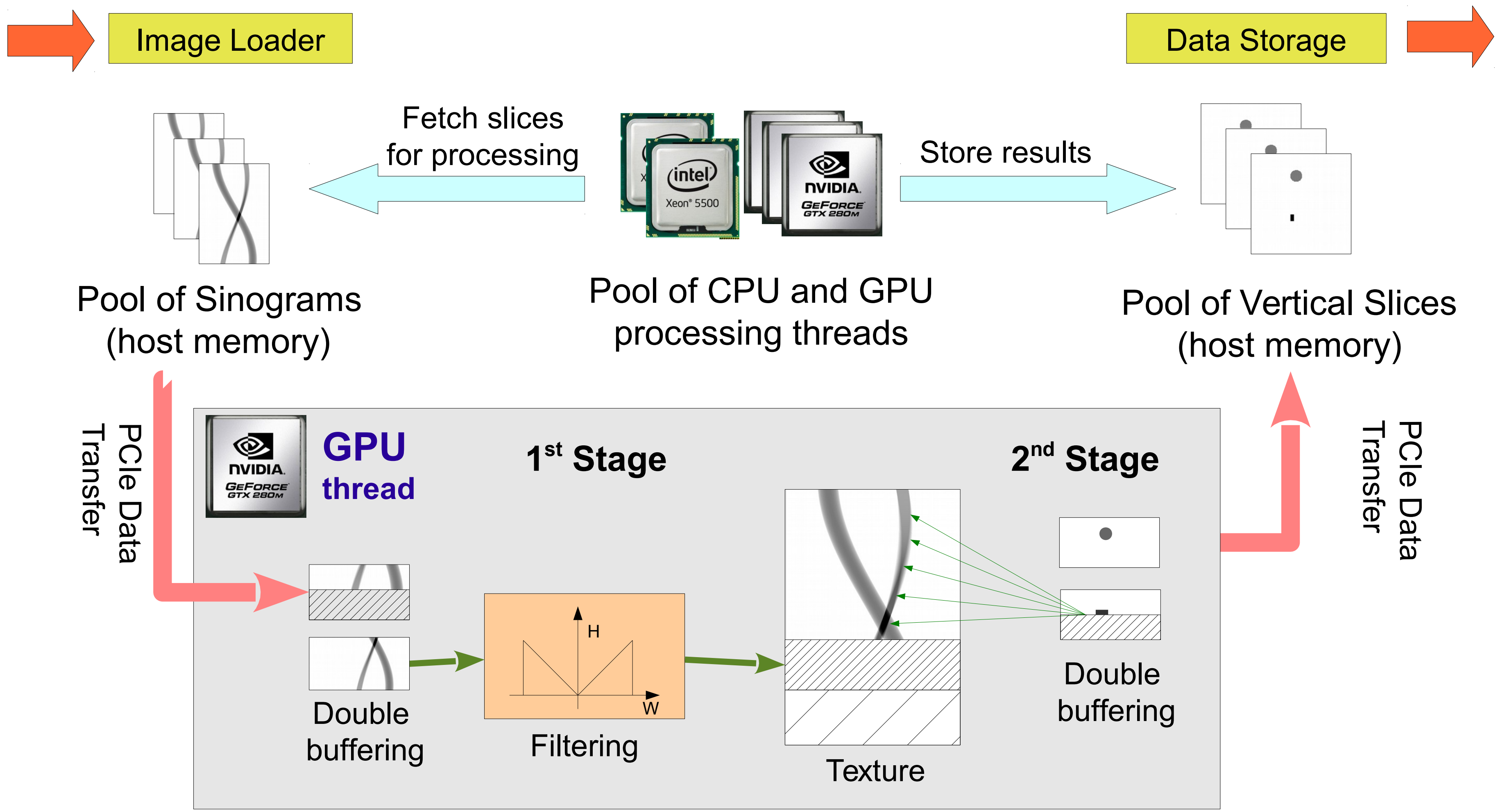
- Linear interpolation, i.e. image scaling
- Optimize random access to multidimensional arrays



S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Filtered Back Projection



S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Performance of Texture Engine

	GT280	GTX580
Core Throughput	930 GF	1580 GF
Texture Fill Rate	48 GT/s	49 GT/s
Ratio	**19.3**	**31.6**

Optimizing FBP for Fermi

16 px

16 px

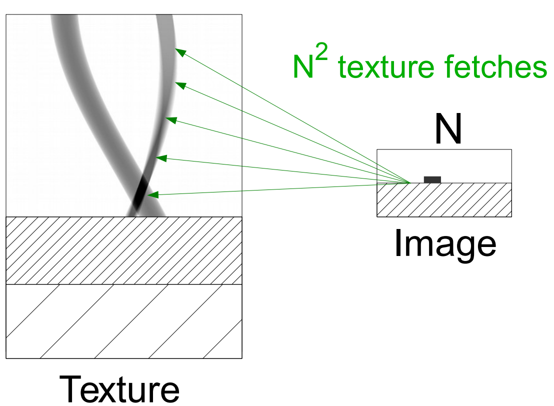
Thread block

$$v = x \bullet \cos(\alpha) - y \bullet \sin(\alpha)$$

$$\max_{x,y<N}(v) - \min_{x,y<N}(v) < N\sqrt{2}$$

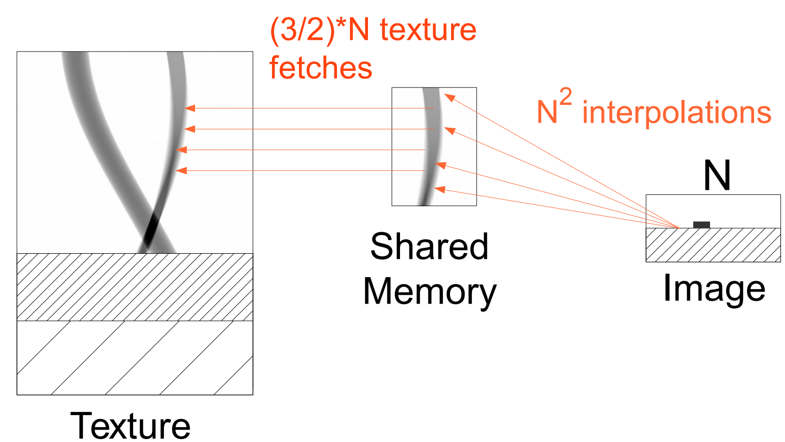
$$N\sqrt{2} < 1.5\,N$$

Each block of threads accesses actually only 3 • N / 2 bins per projection



Standard Version

Texture engine is heavily loaded



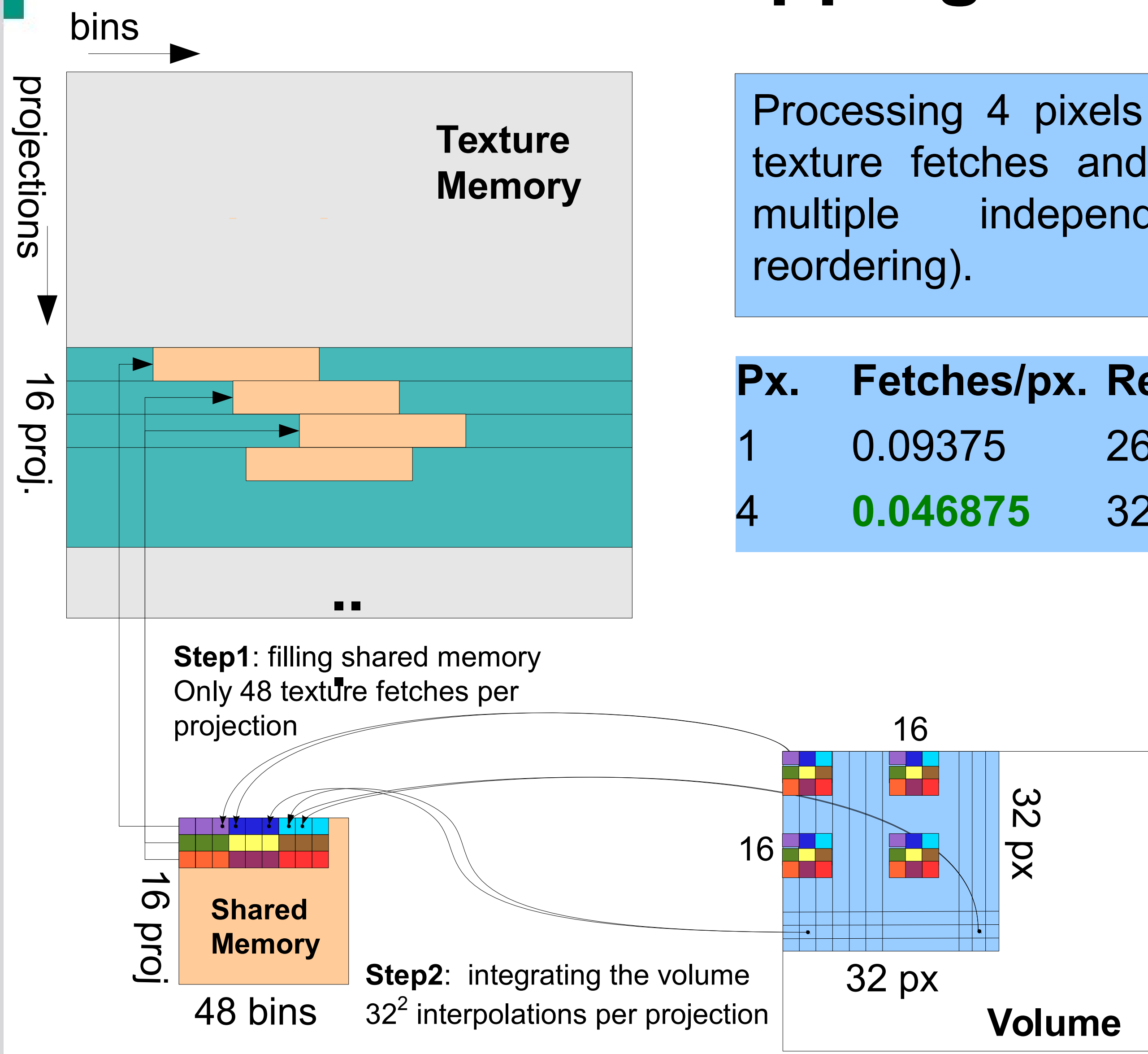
Fermi-optimized Version

Both texture & computations engines are used

S. Chilingaryan, M. Vogelgesang, A. Mirone, A. Kopmann

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

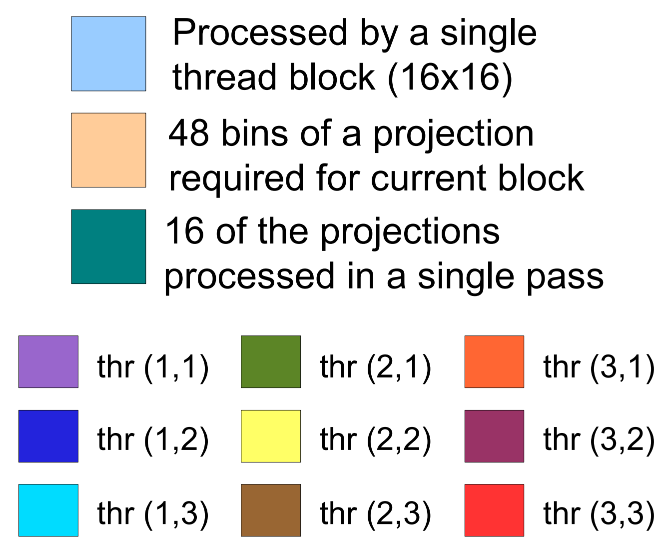
Pixel to thread mapping



Processing 4 pixels per thread reducing amount of texture fetches and hides operation latencies with multiple independent operations (instruction reordering).

Px.	Fetches/px.	Regs	ShMem	Occup.	ILP
1	0.09375	26	1536	66%	1
4	**0.046875**	32	3072	66%	**4**

Legend

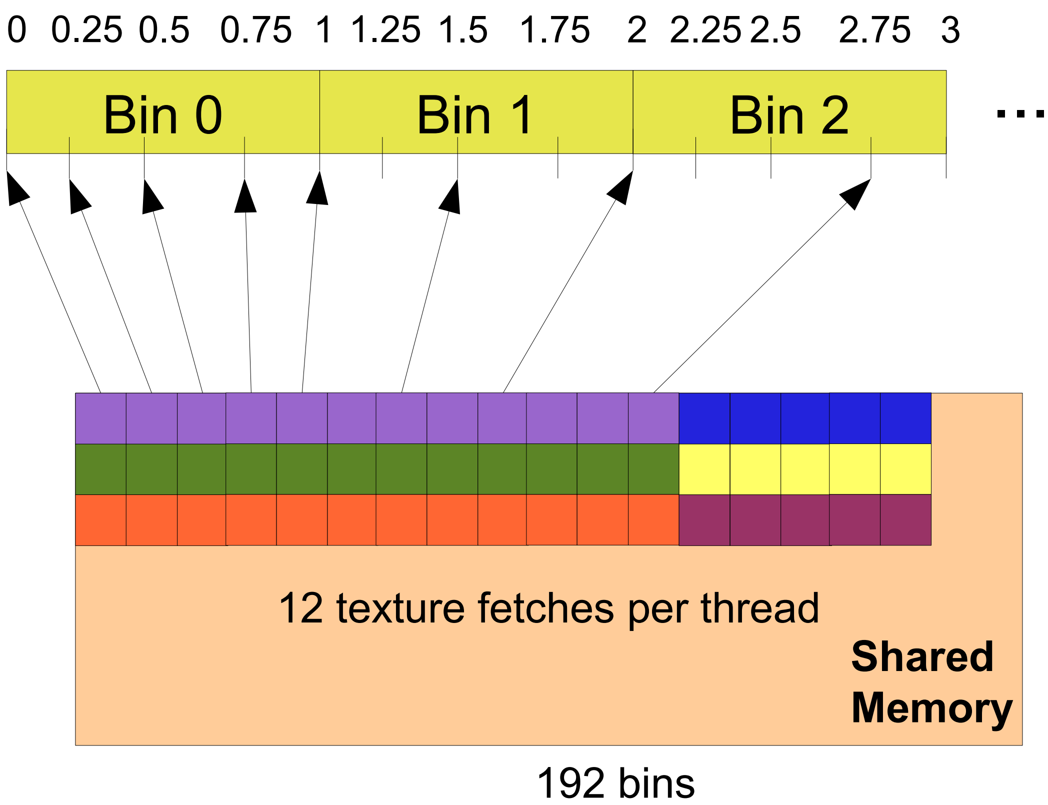


Processing in multiple passes, 16 projections each

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Oversampling



Linear interpolation is slow, and nearest neighbor is not precise enough

With oversampling the texture engine is used to interpolate 4 positions for each projection bin and near-neighbor interpolation is used then.

Method	**Fetches/px**	**Regs**	**ShMem**	**Occup.**	**Reads/px**	**Flops/px**
Linear	0.046875	32	3072	66%	2	7
Oversample	**0.1875**	42	12288	**50%**	**1**	**4**

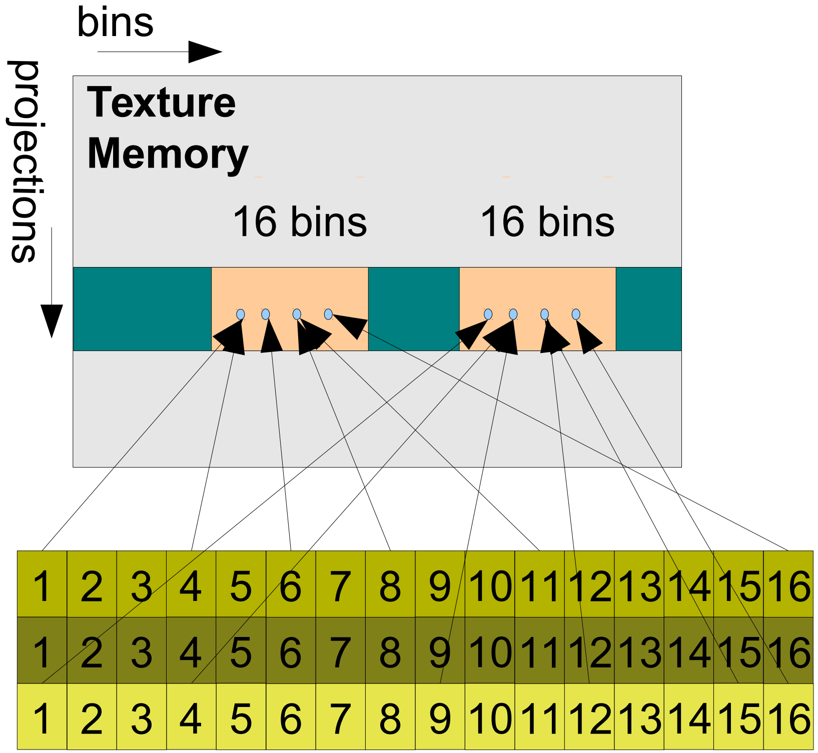
Kepler: Fast Texture Engine is Back

	GT580	GTX680	Change
Texture Engine	49.4 GT/s	128.8 GT/s	**2.6 x**
Floating-point operations	16 x 32 x 1.55 GHz	8 x 192 x 1.006 GHz	**1.94 x**
Integer multiplication, bit operations, type conversions	16 x 16 x 1.55 GHz	8 x 32 x 1.006 GHz	**0.65 x**
Shared Memory	48 KB	48 KB	**1**
Blocks per SM	8	16	**2**
Registers	32K per SM, 63 per thr.	64K per SM, 63 per thr.	**1**

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Default approach



Block of 16x16 pixels

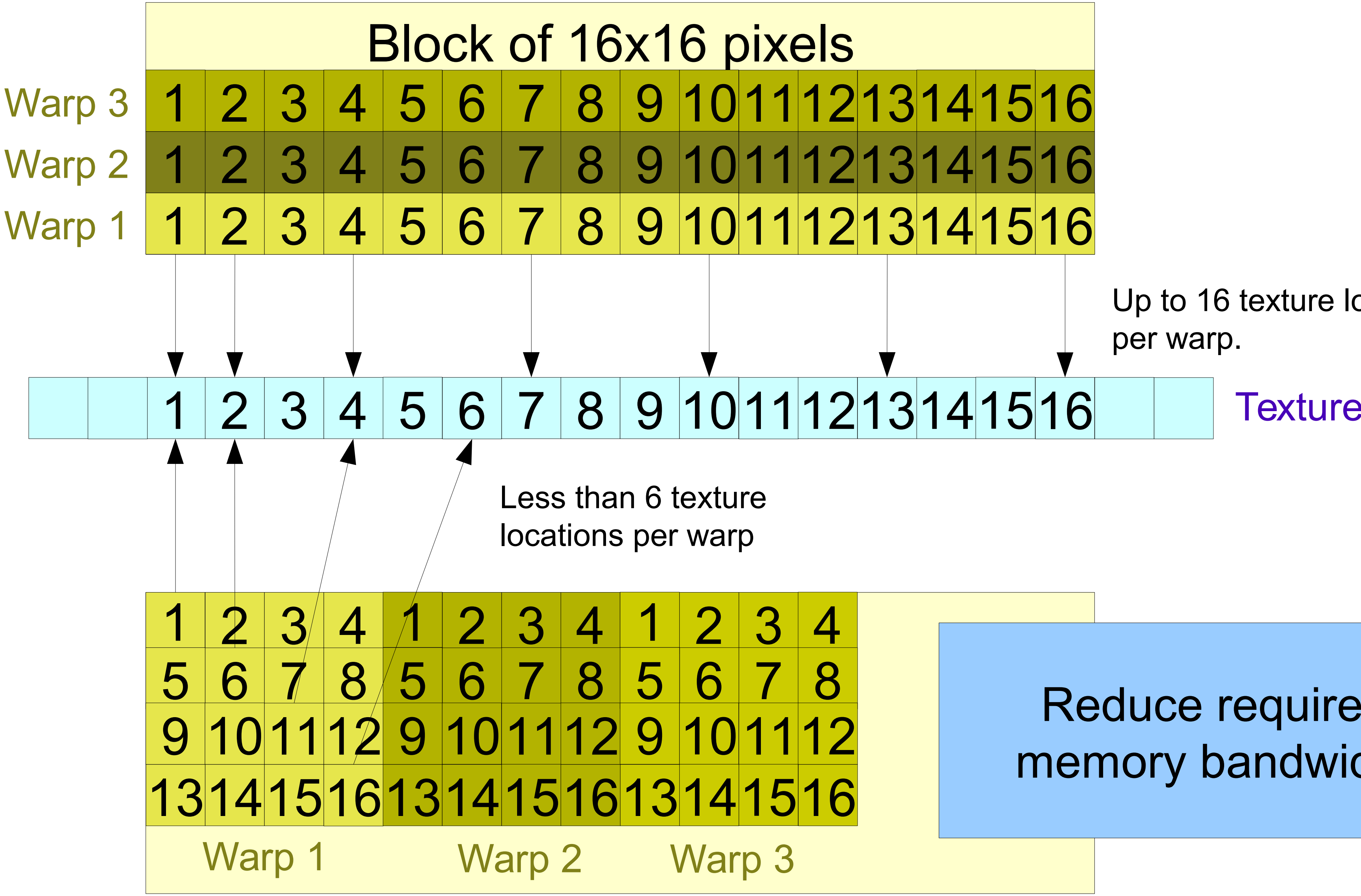
Texture Cache Hit Rate	89 %
Texture Throughput	79.3 GT/s
Theoretical Throughput	128.8 GT/s

1. Up to 16 bins are accessed per warp
2. All threads are accessing a single texture row

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Optimizing the thread mapping



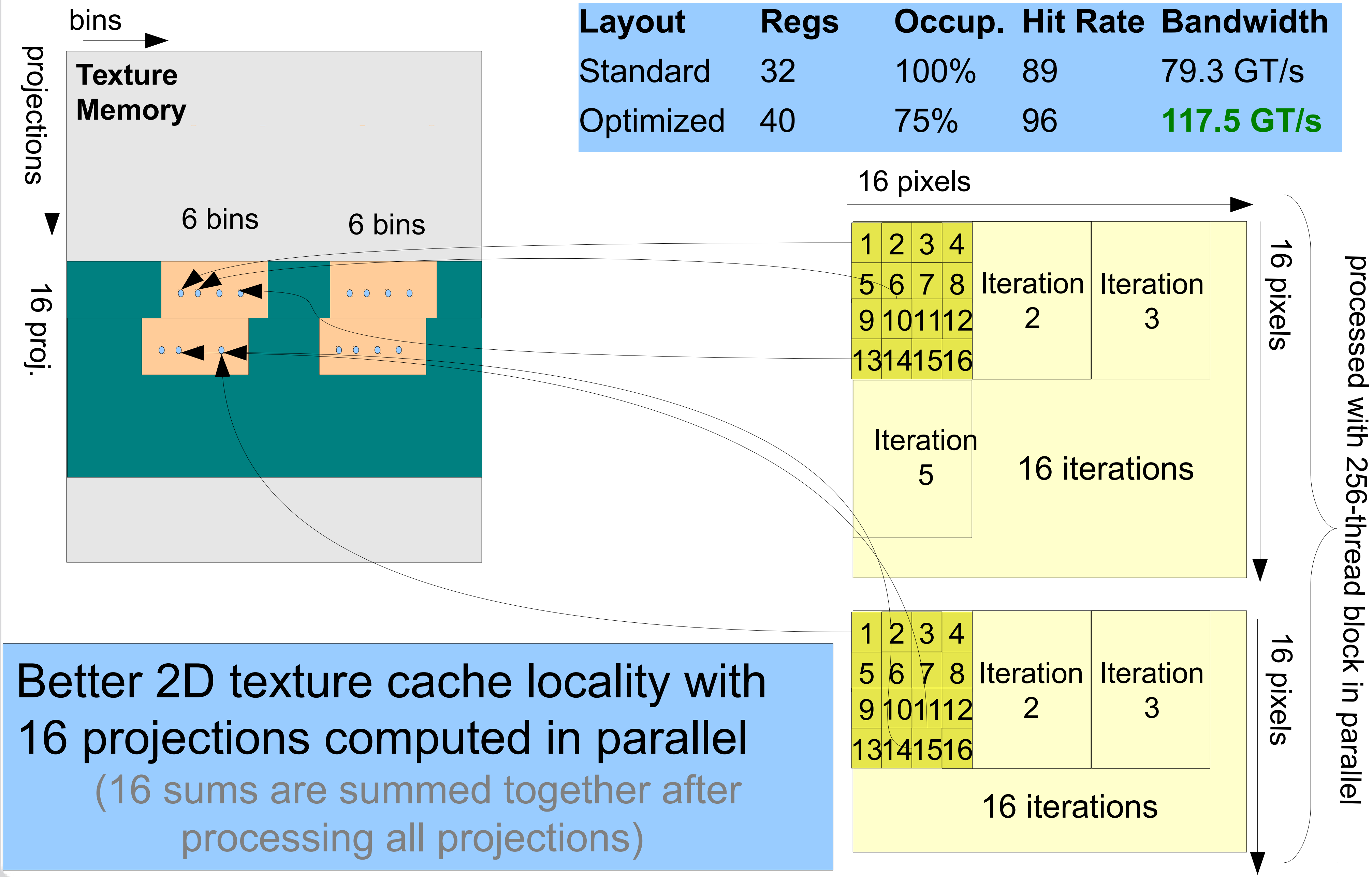
Reduce required memory bandwidth

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Using spatial locality

Layout	Regs	Occup.	Hit Rate	Bandwidth
Standard	32	100%	89	79.3 GT/s
Optimized	40	75%	96	**117.5 GT/s**



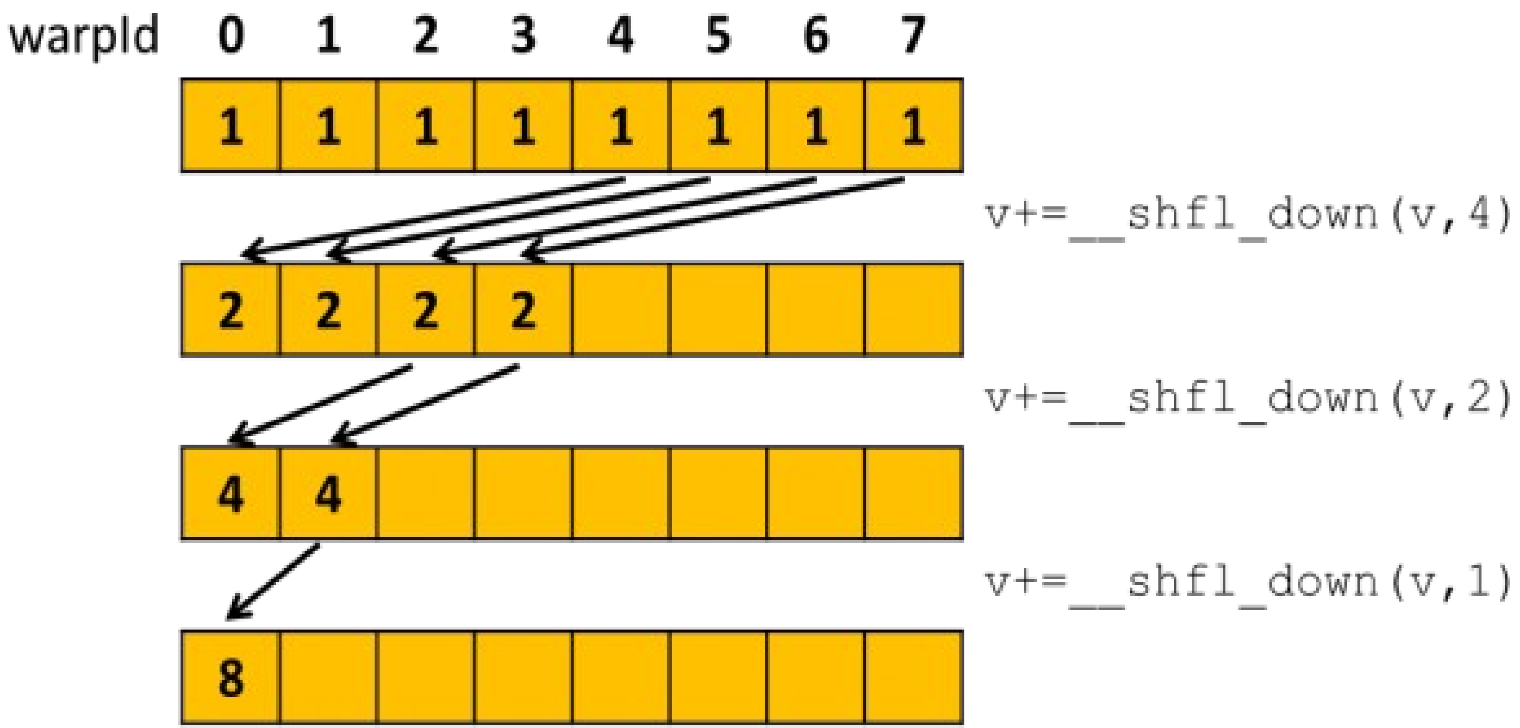
Better 2D texture cache locality with 16 projections computed in parallel

(16 sums are summed together after processing all projections)

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

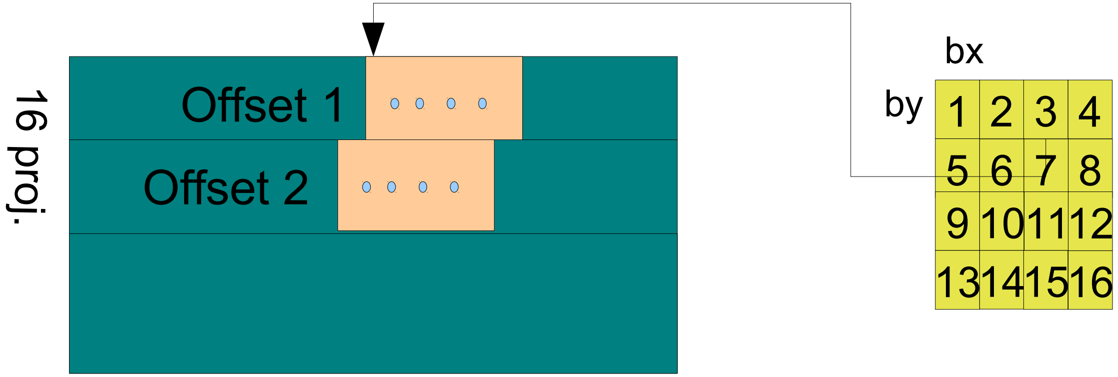
Faster reduction with shuffle instruction



Shuffle instruction introduced by Kepler architecture allows fast exchange of information between threads of the warp.

Oversampling approach on Kepler

Slow performance of integer and rounding operations makes Fermi oversampling algorithm slow.



$$proj\_offset = \lfloor bx \bullet \cos(\alpha) - by \bullet \sin(\alpha) + correction(\alpha) \rfloor$$

On Fermi, for each block and projection we compute smallest-bin offset on the fly by each thread. On Kepler instead we can:

- Optimize rounding routine
- Pre-calculate and cache offsets

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Looking for faster rounding on Kepler

Exponent, 8 bits — Fraction, 23 bits

IEEE 754 single-precision floating point number

$$f = \underbrace{s}\,\underbrace{e\;e\;e\;e\;e\;e\;e\;e}_{\text{Exponent, 8 bits}}\,\underbrace{f\;f\;f\;f\;f\;\ldots\;f\;f\;f\;f}_{\text{Fraction, 23 bits}}$$

31 … 0

$$f = -1^{s} \cdot 2^{e-127} \cdot (1 + \sum f_i \cdot 2^{i-23})$$

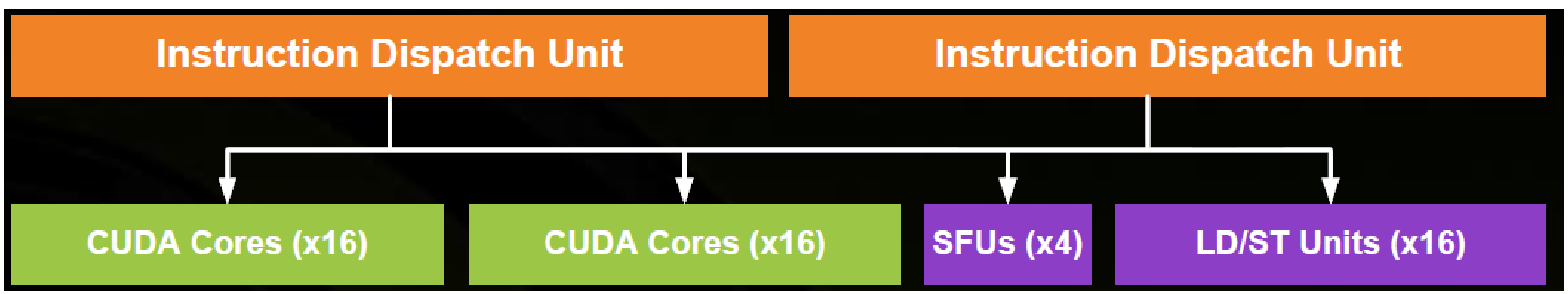
Only 23 significant positions, for small positive numbers:

$$f + 2^{23} = 2^{23} \cdot (1 + \sum f_i \cdot 2^{i-23})$$

i.e. no fractional part

$$\text{round}(f) = f + 2^{23} - 2^{23}$$

$$(\text{int})f = f + 2^{23} - \text{0x4B000000}$$



fp math — rounding — texture

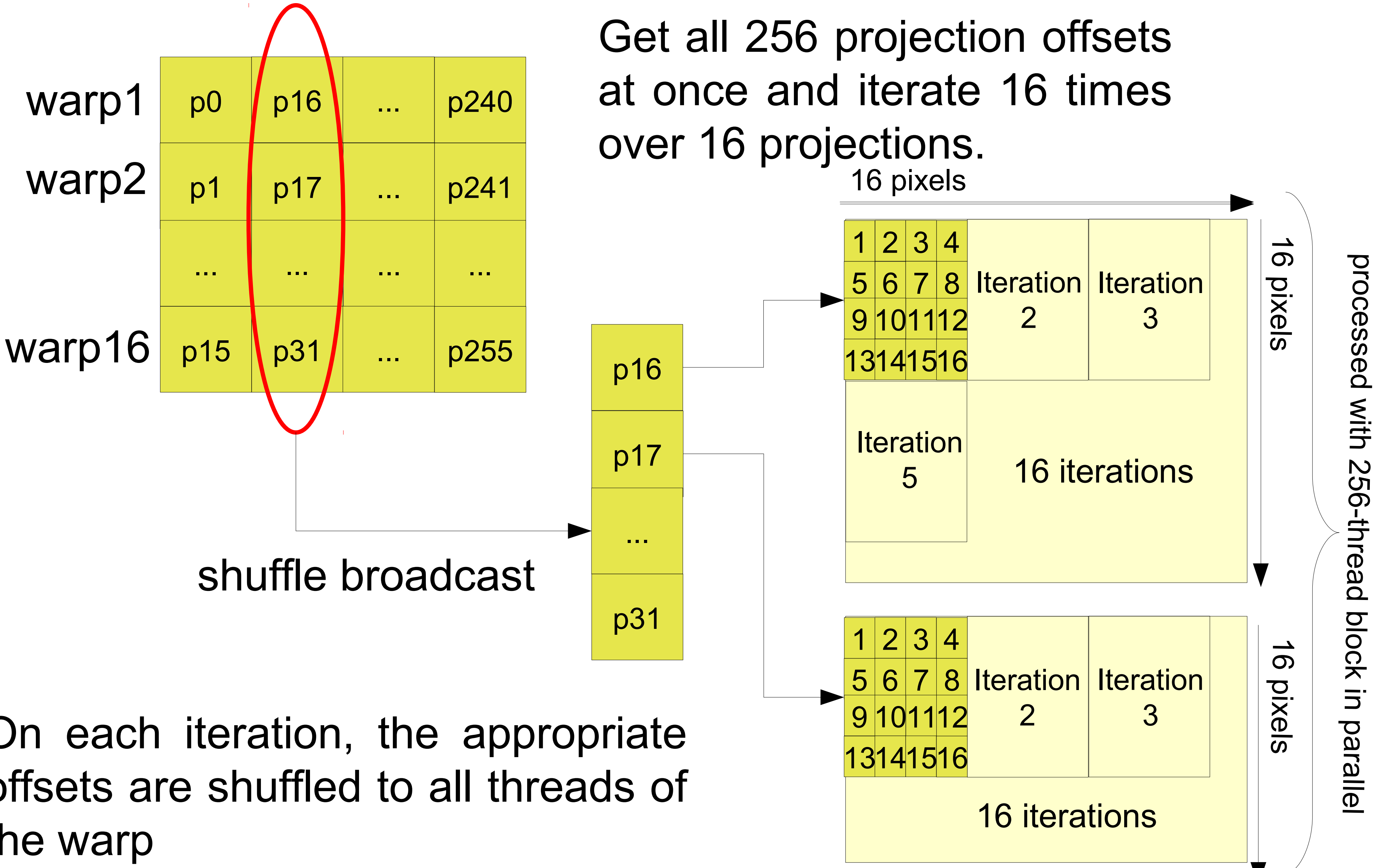
We get faster rounding, but SFUs left unused and we got no speed up...

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Reducing number of rounding operations

Get all 256 projection offsets at once and iterate 16 times over 16 projections.



On each iteration, the appropriate offsets are shuffled to all threads of the warp

S. Chilingaryan et. all

Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Summary: 3 stages of oversampling

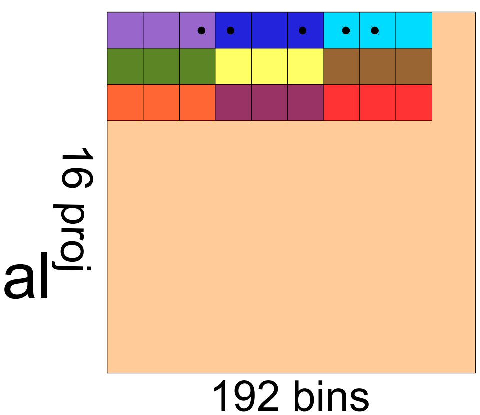
Work-group of 256 threads used to backproject area of 32x32 pixels from 256 projections

1

p0	p16	...	p240
p1	p17	...	p241
...	...	...	...
p15	p31	...	p255

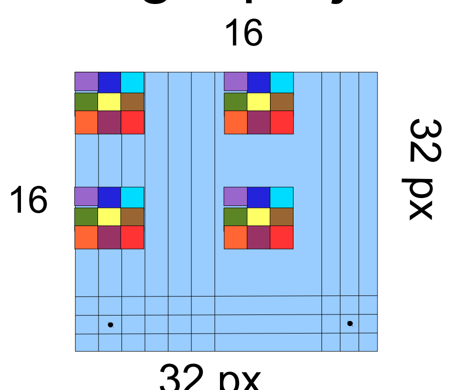
compute all offsets
work-items are mapped linearly to all projections.

2 *16 iterations (only 16 projections at once)*



cache data in shmem
warps are mapped to projections and individual work-items to its bins.

3 256 iterations each processing a single projection



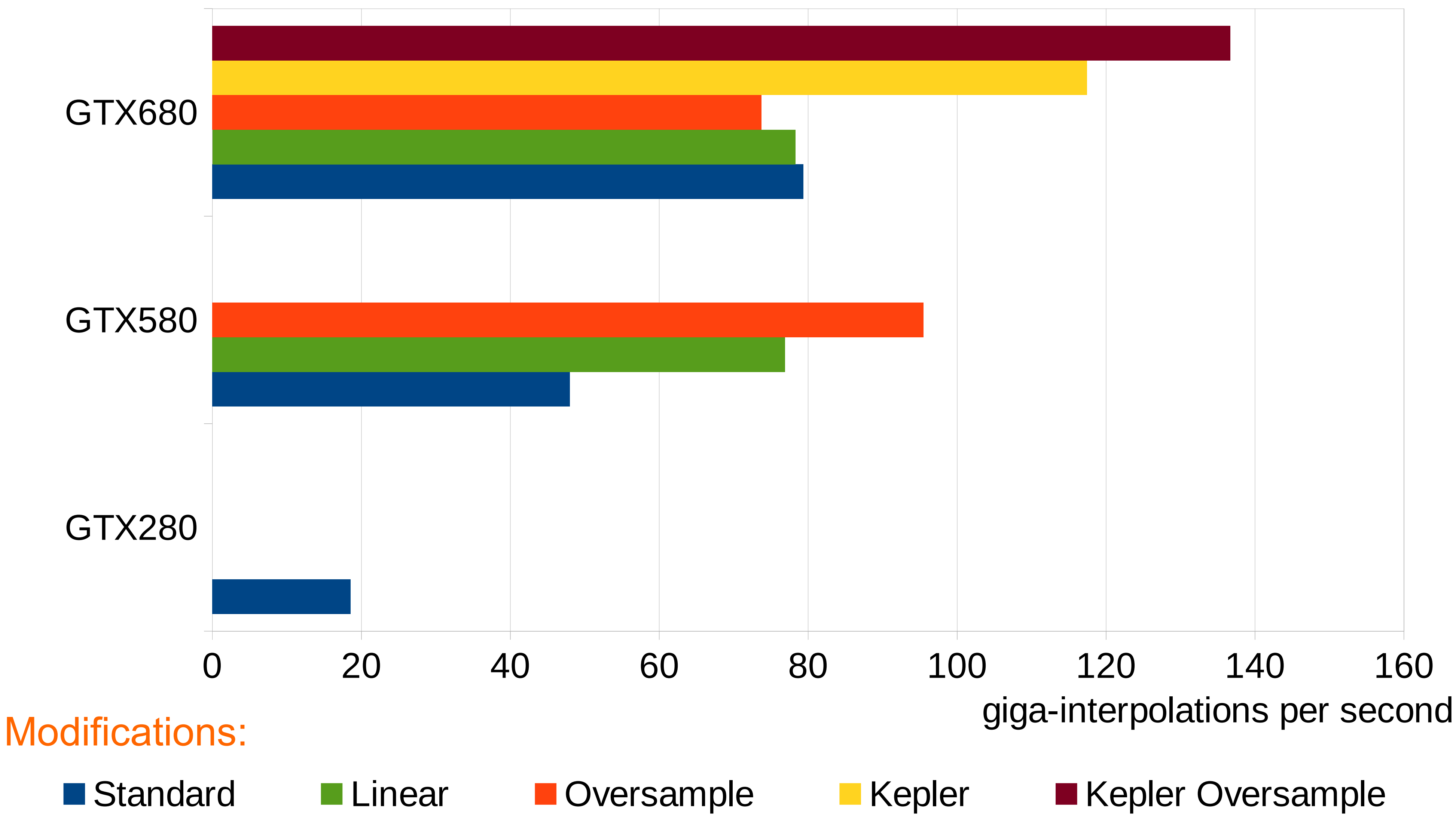
interpolate pixels
work-items are mapped to area 16x16 pixels and proess 4 pixels at once

3 different mappings for optimal performance

S. Chilingaryan et. all

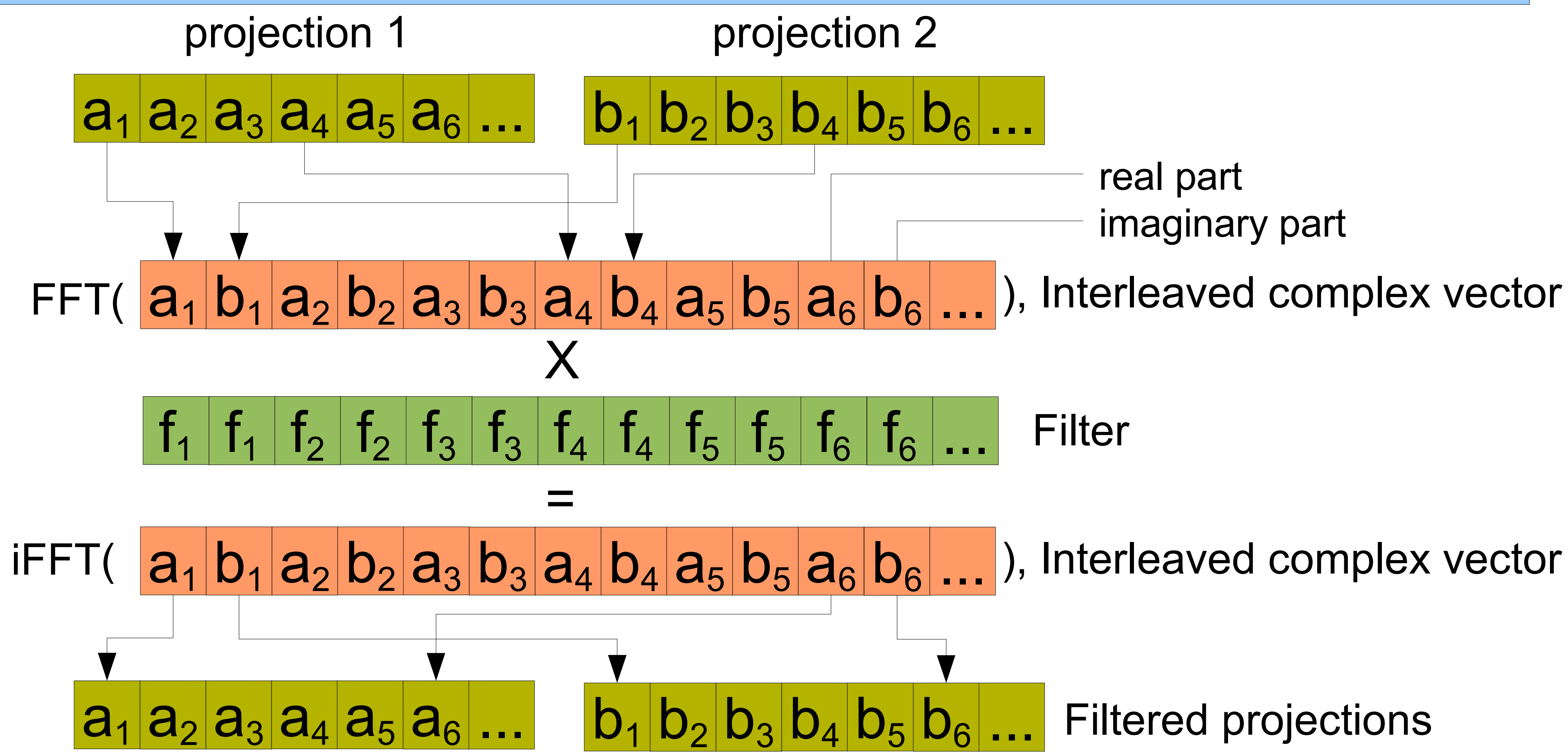
Institute for Data Processing and Electronics
Karlsruhe Institute of Technology

Performance of Back Projection



Optimizing Filtering Step

FFT library is optimized for complex-to-complex transforms while we are dealing with real numbers.



- **Also**
  - Pad data to a size equal to the closest power of 2
  - Batched processing

Summary

- Scalable hardware platform for image-based control
  - Only off-the-shelf components are used
  - Easily scalable from single PC to the GPU cluster
  - Reliable storage for data streaming at rates up to 4 GB/s
  - Distributed over large area using Optical Infiniband Links
- Fully-pipelined parallel image-processing framework
  - Tuning for various parallel architectures
  - Real-time reconstruction (up to 2 GB/s from camera)
  - Fast low-dose reconstruction (about 4 hours per dataset)
- Remote data analysis infrastructure
  - Virtualization environment for remote image segmentation
  - High quality web-based visualization of large volumes