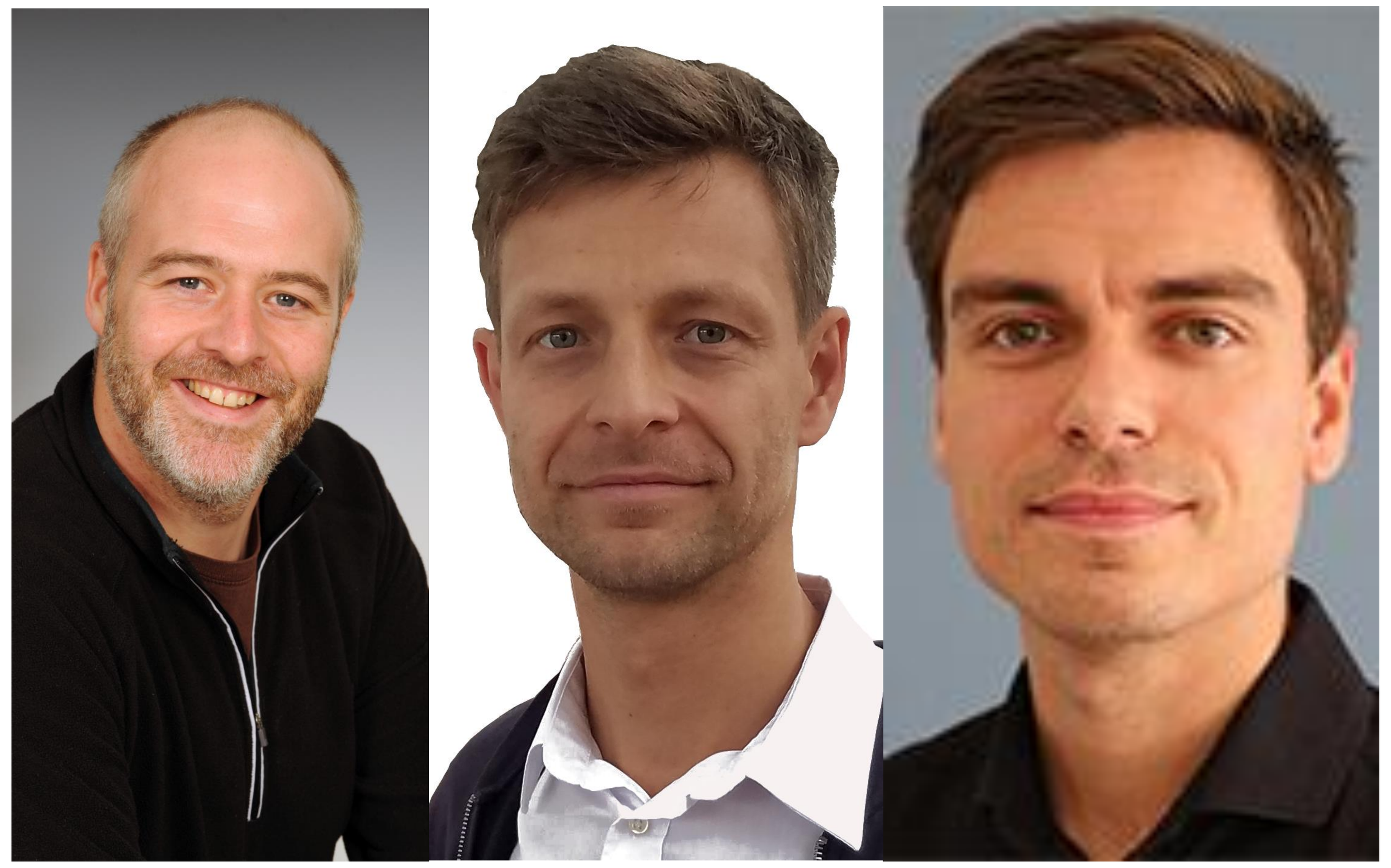
CompBioMed

A Centre of Excellence in Computational Biomedicine

e-Seminar #22

Tools and techniques for making efficient use of GPUs



17 March 2022

The e-Seminar will start at 3pm CET / 2pm GMT

Presenters:

Paul Graham, Robert Dietrich, Felix Schmitt (NVIDIA)

Moderator:

Tim Weaving (University College London)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 823712

INSILICO WORLD

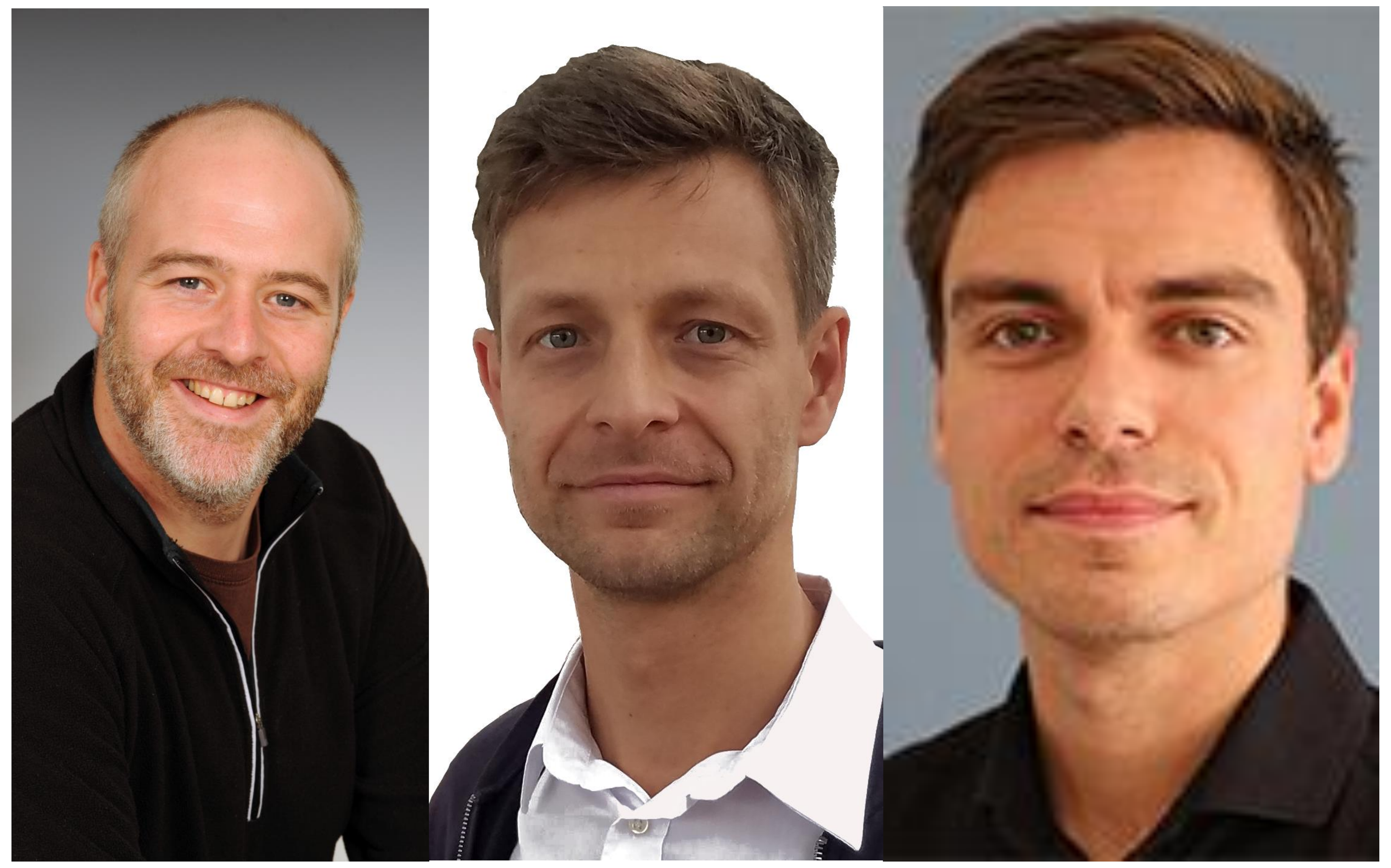
https://insilicoworld.slack.com/archives/C0151M02TA4

The e-Seminar series is run in collaboration with:

VPH Institute

Building the Virtual Physiological Human

e-Seminar #22
Tools and techniques for making efficient use of GPUs



17 March 2022

Welcome!

Presenters:
Paul Graham, Robert Dietrich, Felix Schmitt (NVIDIA)

Moderator:
Tim Weaving (University College London)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 823712

INSILICO WORLD
https://insilicoworld.slack.com/archives/C0151M02TA4

The e-Seminar series is run in collaboration with:
VPH Institute
Building the Virtual Physiological Human

MAKING EFFICIENT USE OF GPUS

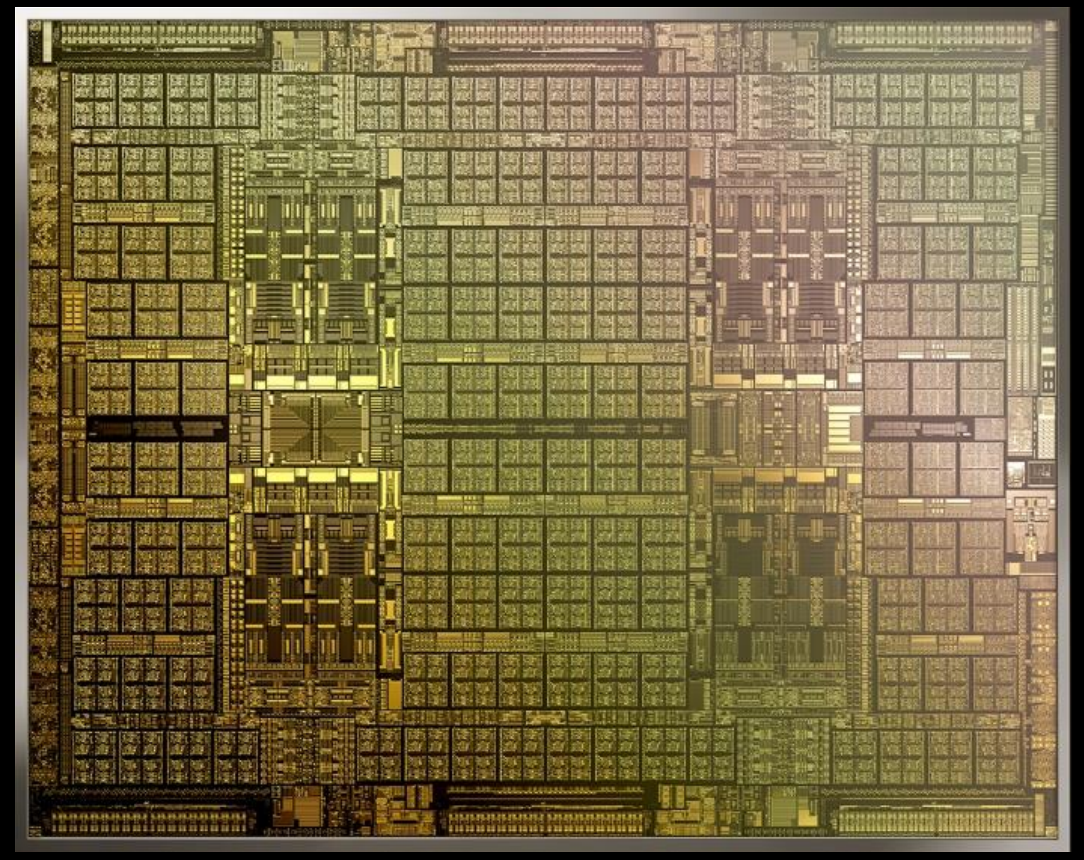
NVIDIA
Paul Graham — Senior Solutions Architect
Robert Dietrich — Senior System Software Engineer
Felix Schmitt — Senior Software Engineer

AGENDA

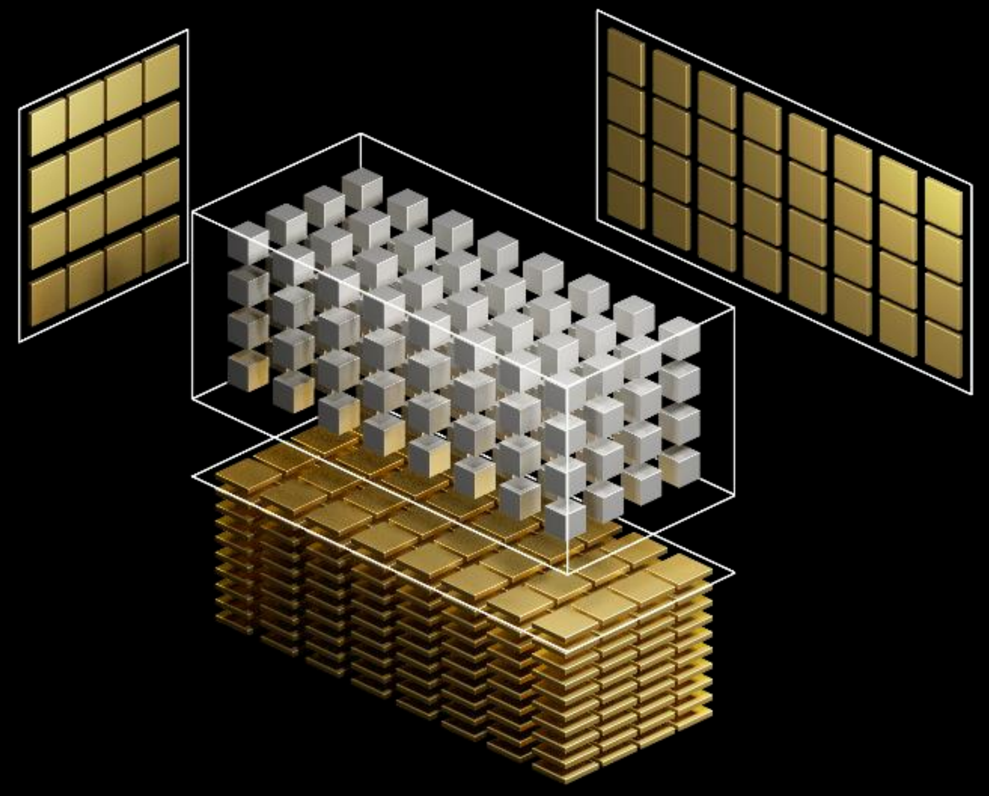
- Technologies & Techniques
  - Programming approaches
  - GPUDirect
  - Visualisation
- Profiling Tools
  - Nsight Systems
  - Nsight Compute

Universe of GPU Computing

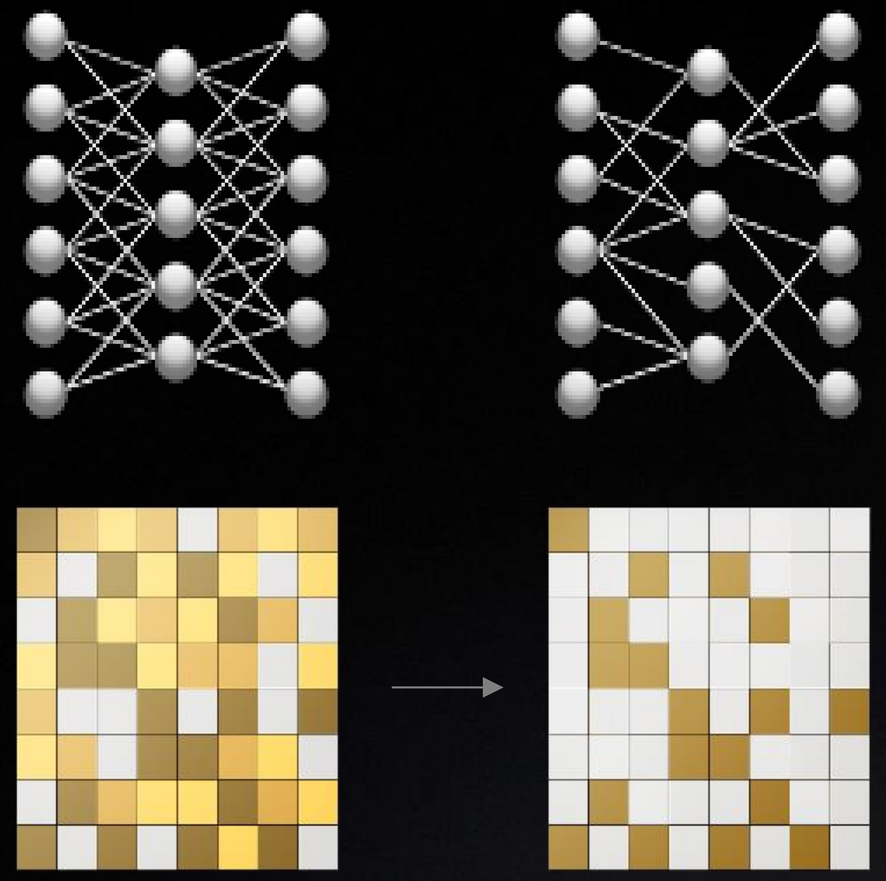
FEATURES OF THE A100



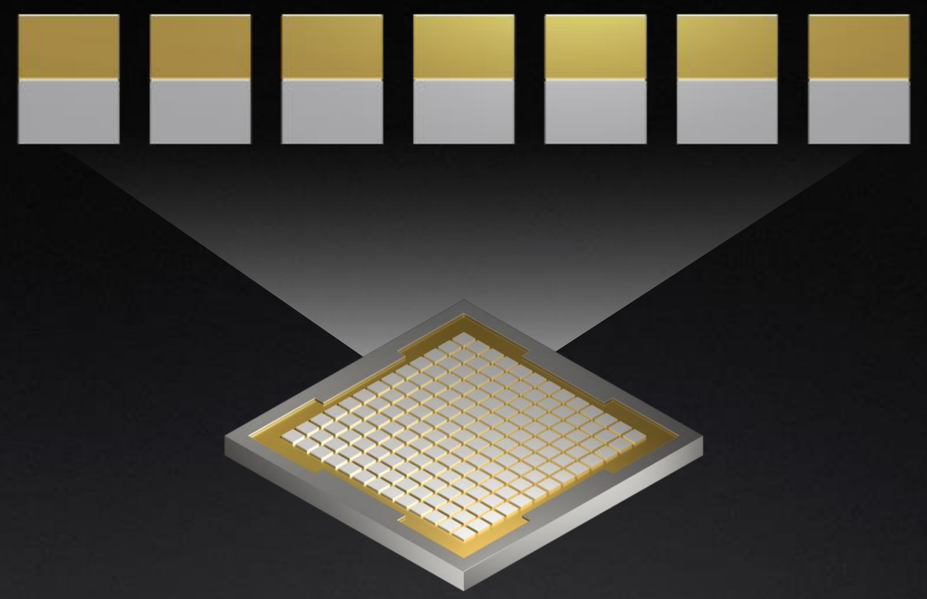
Ampere
World's Largest 7nm chip
54B XTORS, HBM2



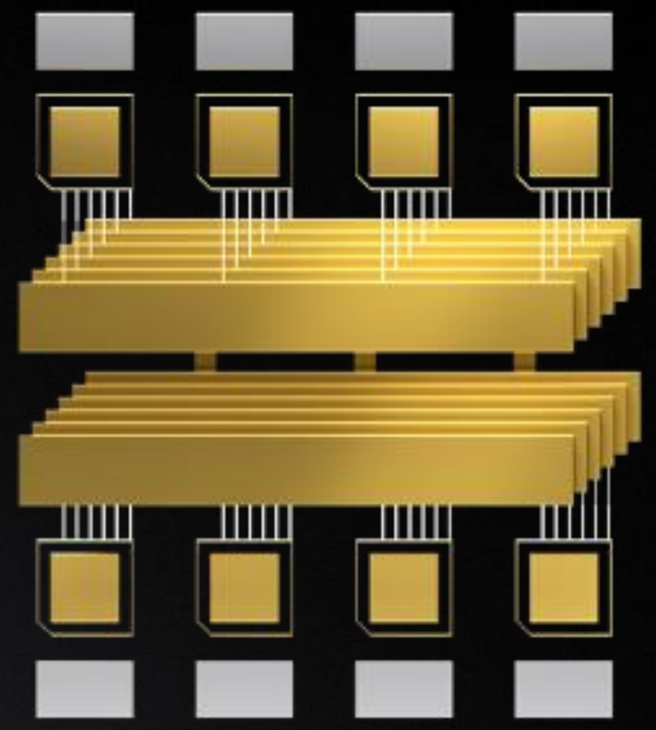
3rd Gen Tensor Cores
Faster, Flexible, Easier to use
20x AI Perf with TF32



New Sparsity Acceleration
Harness Sparsity in AI Models
2x AI Performance



New Multi-Instance GPU
Optimal utilization with right sized GPU
7x Simultaneous Instances per GPU

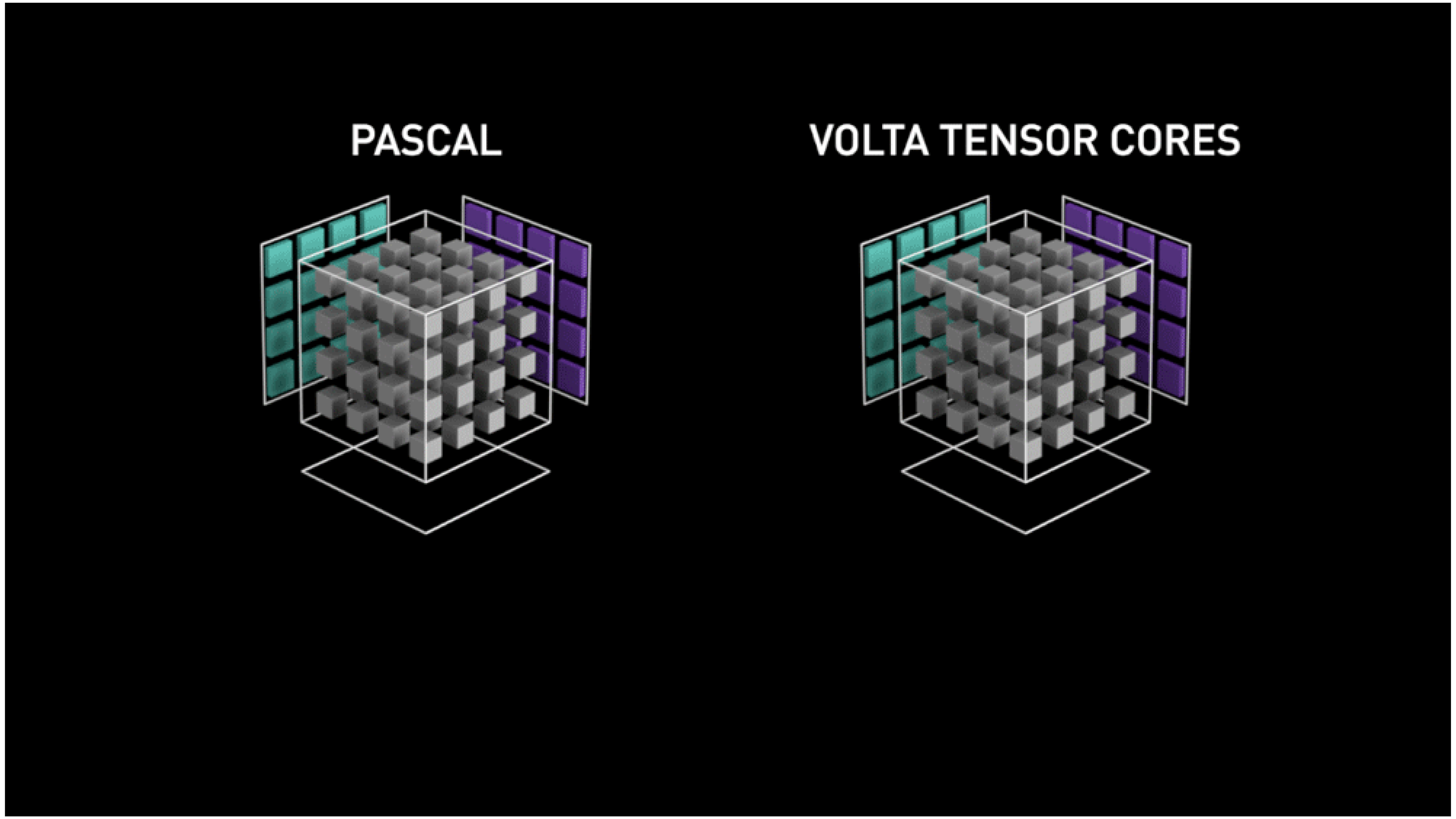


3rd Gen NVLINK and NVSWITCH
Efficient Scaling to Enable Super GPU
2X More Bandwidth

TENSOR CORES

Hardware for Matrix Multiply and Accumulate operations

- Introduced in the V100
- Perform several MMA calcs per clock cycle
    - FP32 in, FP32 out (accumulate)
    - FP16 multiply
- Turing added int8, int4 calculations
- Ampere
    - Full FP64 MMA
    - Bfloat16, Tensor



WAYS TO ACCELERATION

Applications and Frameworks

Libraries	Directives	Programming Languages
“Drop-in” Acceleration	Easily Accelerate Applications	Maximum Flexibility

NVIDIA MATH LIBRARIES

Linear Algebra, FFT, RNG and Basic Math

cuBLAS

cuSPARSE

cuTENSOR

cuSOLVER

CUTLASS

AMGX

cuRAND

cuFFT

CUDA Math API

TENSOR CORE SUPPORT IN MATH LIBRARIES

High-level overview of supported functionality by each library

Library and Tensor Core Functionality	INT4		INT8		FP16		BF16		TF32		FP64
	Dense	Sparse	Dense	Sparse	Dense	Sparse	Dense	Sparse	Dense	Sparse	Dense
cuBLAS & cuBLASLt Dense GEMM			✓		✓		✓		✓		✓
cuTENSOR Tensor Contractions					✓		✓		✓		✓
cuSOLVER Linear System Solvers					✓		✓		✓		✓
cuSPARSE Block-SpMM			✓		✓		✓		✓		✓
cuSPARSELt SpMM				✓		✓		✓		✓	
CUTLASS Dense GEMM and SpMM	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CUTLASS Convolutions	✓		✓		✓		✓		✓		

MULTI-NODE MATH LIBRARIES

cuSOLVERMp: Dense Linear Algebra at scale

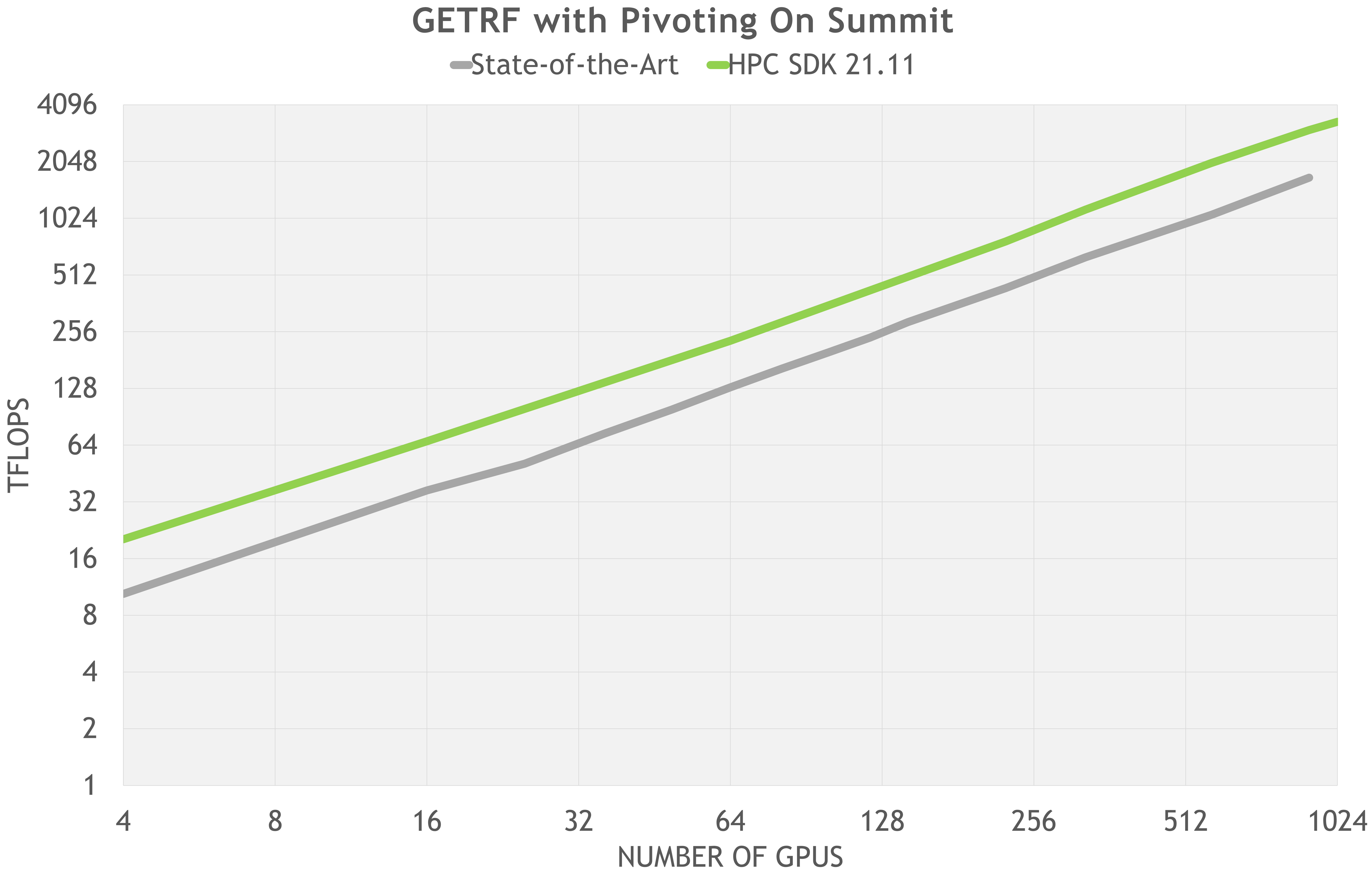
cuSOLVERMp

A distributed-memory multi-node and multiGPU solution for solving systems of linear equations at scale.

GA release available in HPC SDK 21.11

Initial release to support LU Decomposition, with and without pivoting, and Cholesky.

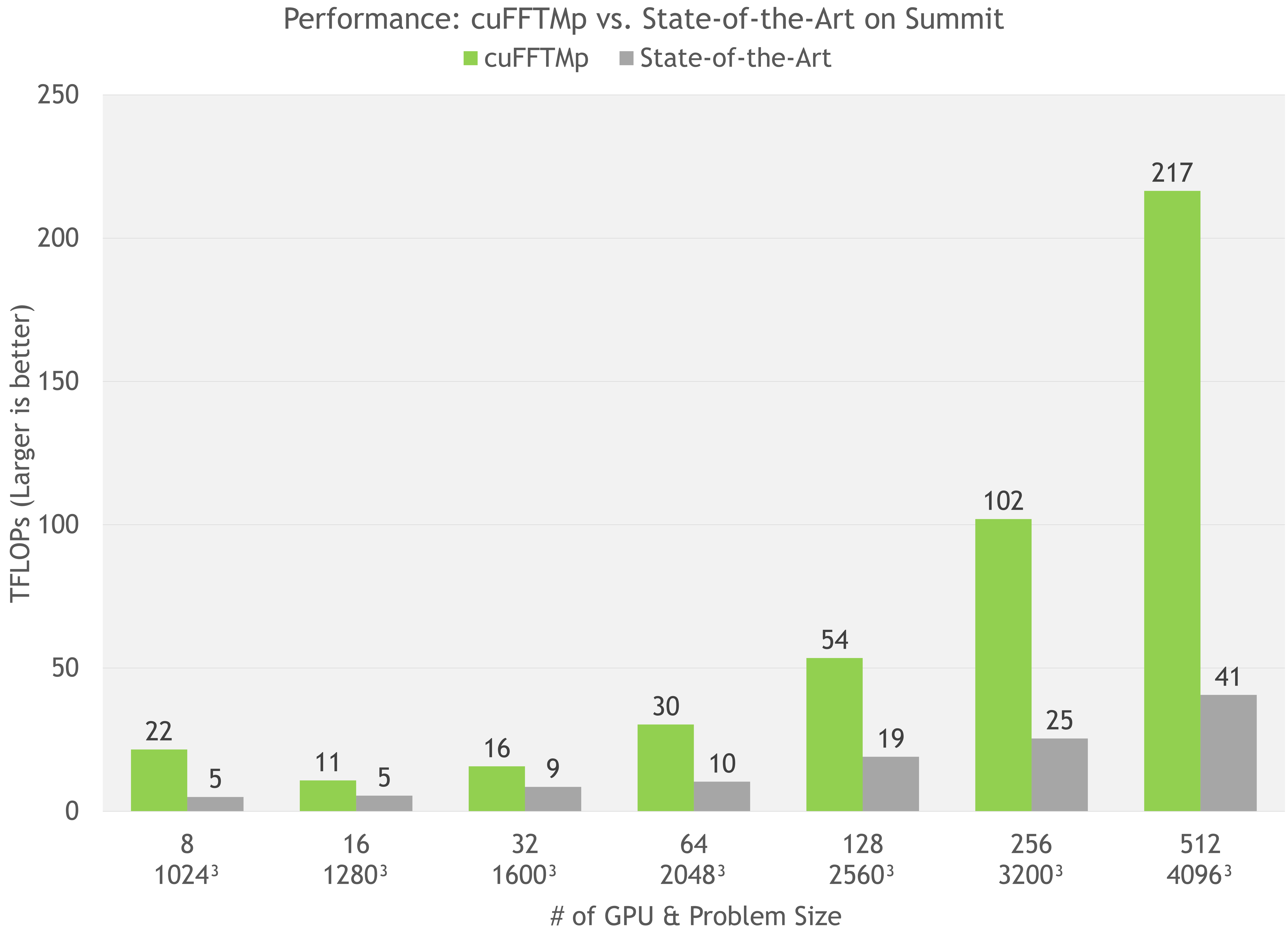
Multiple RHS coming soon!



* Problem size is increased with number of GPUs

MULTI-NODE MATH LIBRARIES

cuFFTMp: Fast Fourier Transforms at scale



cuFFTMp

A distributed-memory multi-node and multiGPU solution for solving FFTs at scale.

EA release available in Autumn '21
https://developer.nvidia.com/cudamathlibraryea

Initial release to 2D & 3D with Slab composition

PROGRAMMING THE NVIDIA PLATFORM

CPU, GPU, and Network

ACCELERATED STANDARD LANGUAGES

ISO C++, ISO Fortran

```
std::transform(par, x, x+n, y, y,
    [=](float x, float y){ return y +
a*x; }
);
```

```
do concurrent (i = 1:n)
   y(i) = y(i) + a*x(i)
enddo
```

```
import cunumeric as np
…
def saxpy(a, x, y):
    y[:] += a*x
```

INCREMENTAL PORTABLE OPTIMIZATION

OpenACC, OpenMP

```
#pragma acc data copy(x,y) {
...
std::transform(par, x, x+n, y, y,
    [=](float x, float y){
        return y + a*x;
});
...
}
```

```
#pragma omp target data map(x,y) {
...
std::transform(par, x, x+n, y, y,
    [=](float x, float y){
        return y + a*x;
});
...
}
```

PLATFORM SPECIALIZATION

CUDA

```
__global__
void saxpy(int n, float a,
           float *x, float *y) {
  int i = blockIdx.x*blockDim.x +
          threadIdx.x;
  if (i < n) y[i] += a*x[i];
}

int main(void) {
  ...
  cudaMemcpy(d_x, x, ...);
  cudaMemcpy(d_y, y, ...);

  saxpy<<<(N+255)/256,256>>>(...);

  cudaMemcpy(y, d_y, ...);
```

ACCELERATION LIBRARIES

Core	Math	Communication	Data Analytics	AI	Quantum

NVIDIA HPC SDK

Available at developer.nvidia.com/hpc-sdk, on NGC, via Spack, and in the Cloud

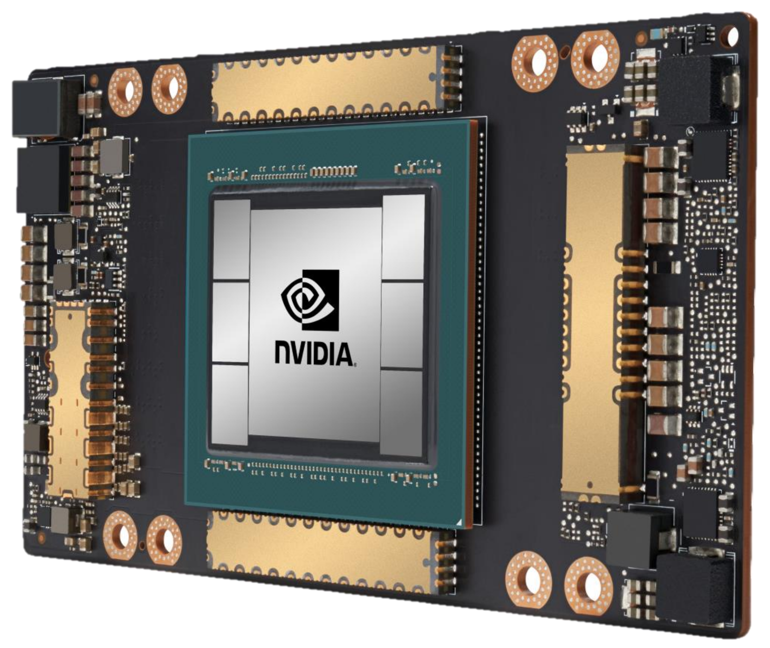
Develop for the NVIDIA Platform: GPU, CPU and Interconnect

Libraries | Accelerated C++ and Fortran | Directives | CUDA

7-8 Releases Per Year | Freely Available

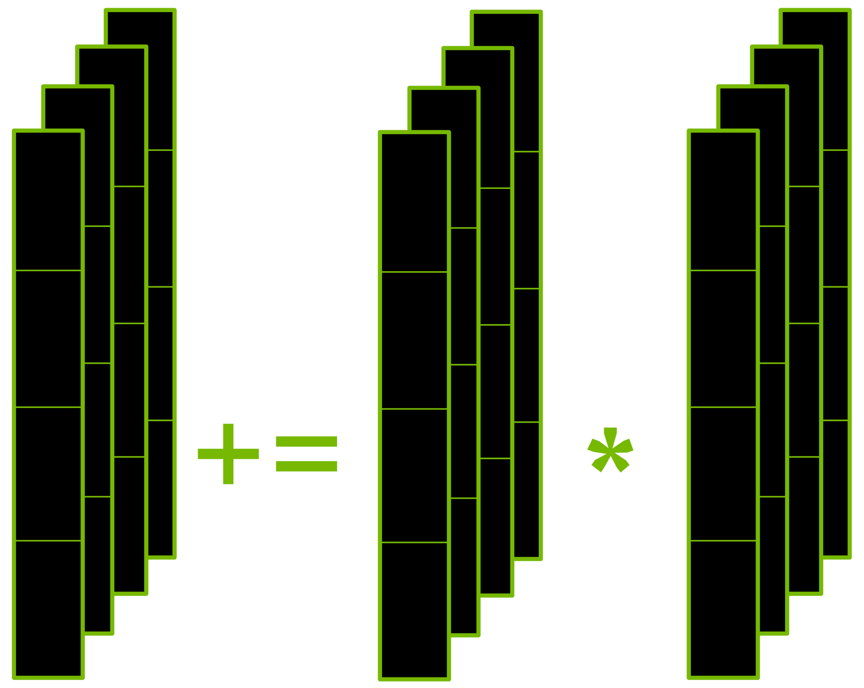
HPC COMPILERS

NVC | NVC++ | NVFORTRAN

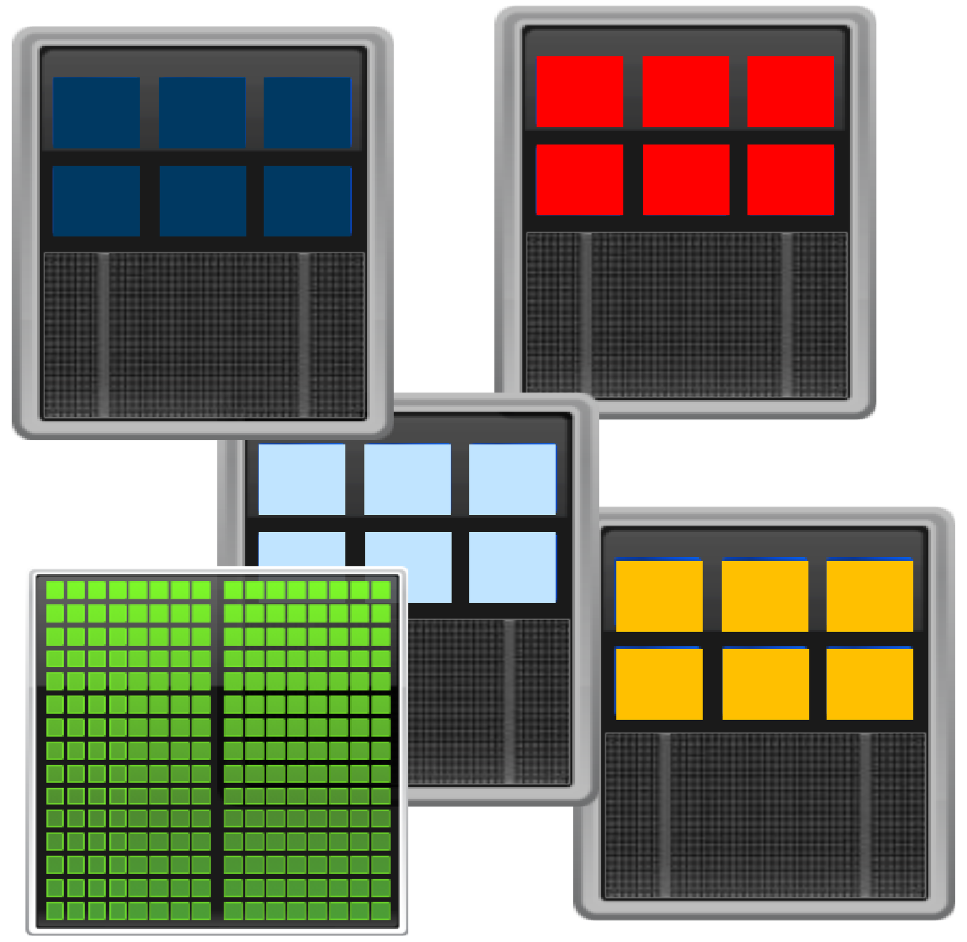


Accelerated
A100

Programmable
Standard Languages
Directives
CUDA



CPU Optimized
Directives
Vectorization



Multi-Platform
x86_64
Arm
OpenPOWER

```cpp
static inline
void CalcHydroConstraintForElems(Domain &domain, Index_t length,
    Index_t *regElemlist, Real_t dvovmax, Real_t& dthydro)
{
#if _OPENMP
  const Index_t threads = omp_get_max_threads();
  Index_t hydro_elem_per_thread[threads];
  Real_t dthydro_per_thread[threads];
#else
  Index_t threads = 1;
  Index_t hydro_elem_per_thread[1];
  Real_t dthydro_per_thread[1];
#endif
#pragma omp parallel firstprivate(length, dvovmax)
  {
    Real_t dthydro_tmp = dthydro ;
    Index_t hydro_elem = -1 ;
#if _OPENMP
    Index_t thread_num = omp_get_thread_num();
#else
    Index_t thread_num = 0;
#endif
#pragma omp for
    for (Index_t i = 0 ; i < length ; ++i) {
      Index_t indx = regElemlist[i] ;

      if (domain.vdov(indx) != Real_t(0.)) {
        Real_t dtdvov = dvovmax / (FABS(domain.vdov(indx))+Real_t(1.e-20)) ;

        if ( dthydro_tmp > dtdvov ) {
          dthydro_tmp = dtdvov ;
          hydro_elem = indx ;
        }
      }
    }
    dthydro_per_thread[thread_num] = dthydro_tmp ;
    hydro_elem_per_thread[thread_num] = hydro_elem ;
  }
  for (Index_t i = 1; i < threads; ++i) {
    if(dthydro_per_thread[i] < dthydro_per_thread[0]) {
      dthydro_per_thread[0] = dthydro_per_thread[i];
      hydro_elem_per_thread[0] = hydro_elem_per_thread[i];
    }
  }
  if (hydro_elem_per_thread[0] != -1) {
    dthydro = dthydro_per_thread[0] ;
  }
  return ;
}
```

C++ with OpenMP

STANDARD C++

- Composable, compact and elegant
- Easy to read and maintain
- ISO Standard
- Portable - nvc++, g++, icpc, MSVC, …

```cpp
static inline
void CalcHydroConstraintForElems(Domain &domain, Index_t length,
    Index_t *regElemlist, Real_t dvovmax, Real_t &dthydro)
{
  dthydro = std::transform_reduce(
    std::execution::par, counting_iterator(0), counting_iterator(length),
    dthydro, [](Real_t a, Real_t b) { return a < b ? a : b; },
    [=, &domain](Index_t i)
  {
      Index_t indx = regElemlist[i];
      if (domain.vdov(indx) == Real_t(0.0)) {
        return std::numeric_limits<Real_t>::max();
      } else {
        return dvovmax / (std::abs(domain.vdov(indx)) + Real_t(1.e-20));
      }
  });
}
```

Standard C++

C++ STANDARD PARALLELISM

Lulesh Performance

NVIDIA

HPC PROGRAMMING IN ISO FORTRAN

ISO is the place for portable concurrency and parallelism

Fortran 2018

Array Syntax and Intrinsics

- NVFORTRAN 20.5
- Accelerated matmul, reshape, spread, …

DO CONCURRENT

- NVFORTRAN 20.11
- Auto-offload & multi-core

Co-Arrays

- Coming Soon
- Accelerated co-array images

Preview support available now in NVFORTRAN

Fortran 202x

DO CONCURRENT Reductions

- NVFORTRAN 21.11
- `REDUCE` subclause added
- Support for `+`, `*`, `MIN`, `MAX`, `IAND`, `IOR`, `IEOR`.
- Support for `.AND.`, `.OR.`, `.EQV.`, `.NEQV` on `LOGICAL` values
- Atomics

ACCELERATED PROGRAMMING IN ISO FORTRAN

NVFORTRAN Accelerates Fortran Intrinsics with cuTENSOR Backend

```
real(8), dimension(ni,nk) :: a
real(8), dimension(nk,nj) :: b
real(8), dimension(ni,nj) :: c
...
!$acc enter data copyin(a,b,c) create(d)

do nt = 1, ntimes
  !$acc kernels
  do j = 1, nj
    do i = 1, ni
      d(i,j) = c(i,j)
      do k = 1, nk
        d(i,j) = d(i,j) + a(i,k) * b(k,j)
      end do
    end do
  end do
  !$acc end kernels
end do
!$acc exit data copyout(d)
```

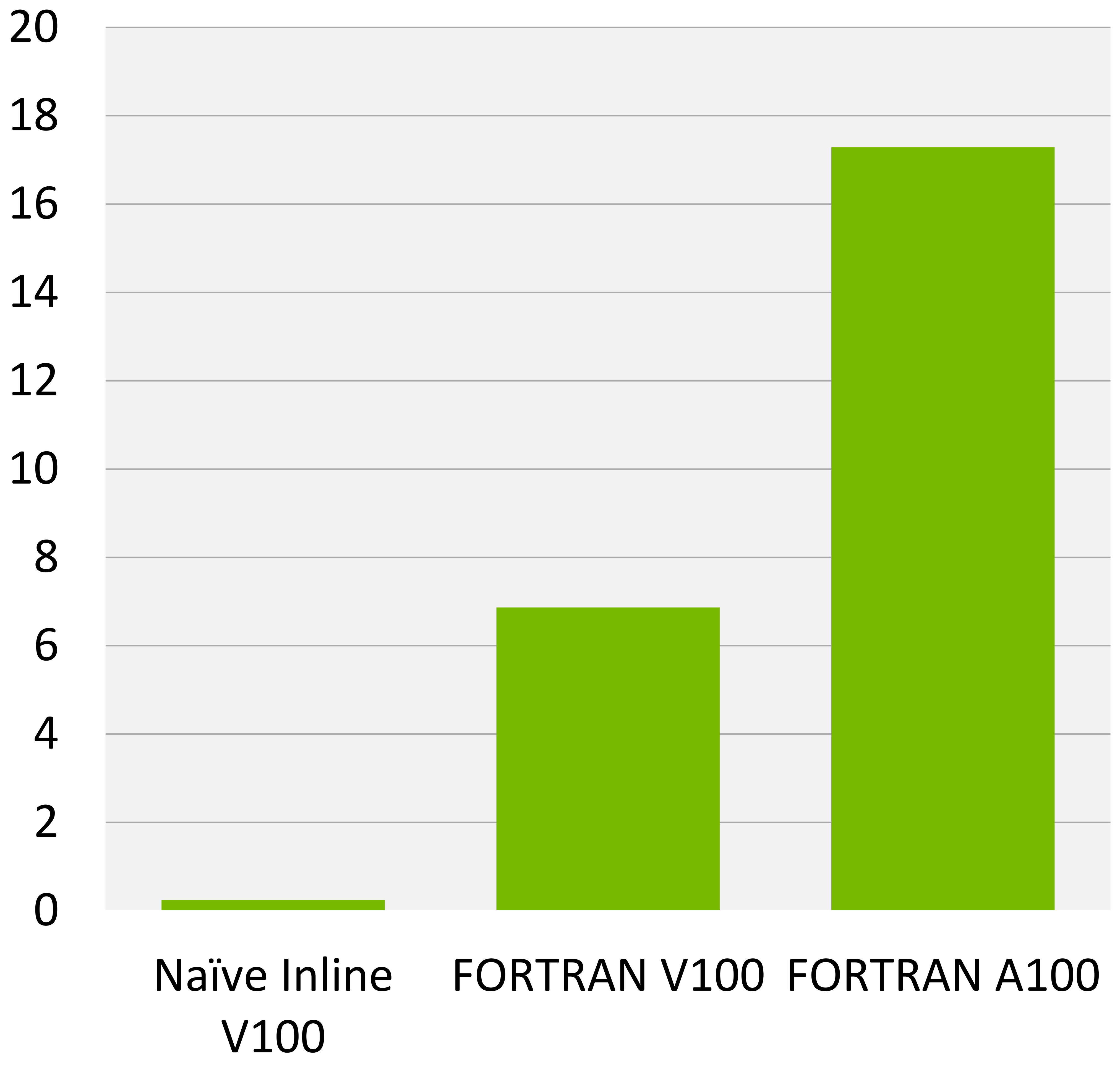
Inline FP64 matrix multiply

```
real(8), dimension(ni,nk) :: a
real(8), dimension(nk,nj) :: b
real(8), dimension(ni,nj) :: c

...

do nt = 1, ntimes
  d = c + matmul(a,b)
end do
```

MATMUL FP64 matrix multiply



HPC PROGRAMMING IN ISO FORTRAN

Examples of Patterns Accelerated in NVFORTRAN

```
d = 2.5 * ceil(transpose(a)) + 3.0 * abs(transpose(b))
d = 2.5 * ceil(transpose(a)) + 3.0 * abs(b)
d = reshape(a,shape=[ni,nj,nk])
d = reshape(a,shape=[ni,nk,nj])
d = 2.5 * sqrt(reshape(a,shape=[ni,nk,nj],order=[1,3,2]))
d = alpha * conjg(reshape(a,shape=[ni,nk,nj],order=[1,3,2]))
d = reshape(a,shape=[ni,nk,nj],order=[1,3,2])
d = reshape(a,shape=[nk,ni,nj],order=[2,3,1])
d = reshape(a,shape=[ni*nj,nk])
d = reshape(a,shape=[nk,ni*nj],order=[2,1])
d = reshape(a,shape=[64,2,16,16,64],order=[5,2,3,4,1])
d = abs(reshape(a,shape=[64,2,16,16,64],order=[5,2,3,4,1]))
c = matmul(a,b)
c = matmul(transpose(a),b)
c = matmul(reshape(a,shape=[m,k],order=[2,1]),b)
c = matmul(a,transpose(b))
c = matmul(a,reshape(b,shape=[k,n],order=[2,1]))
```

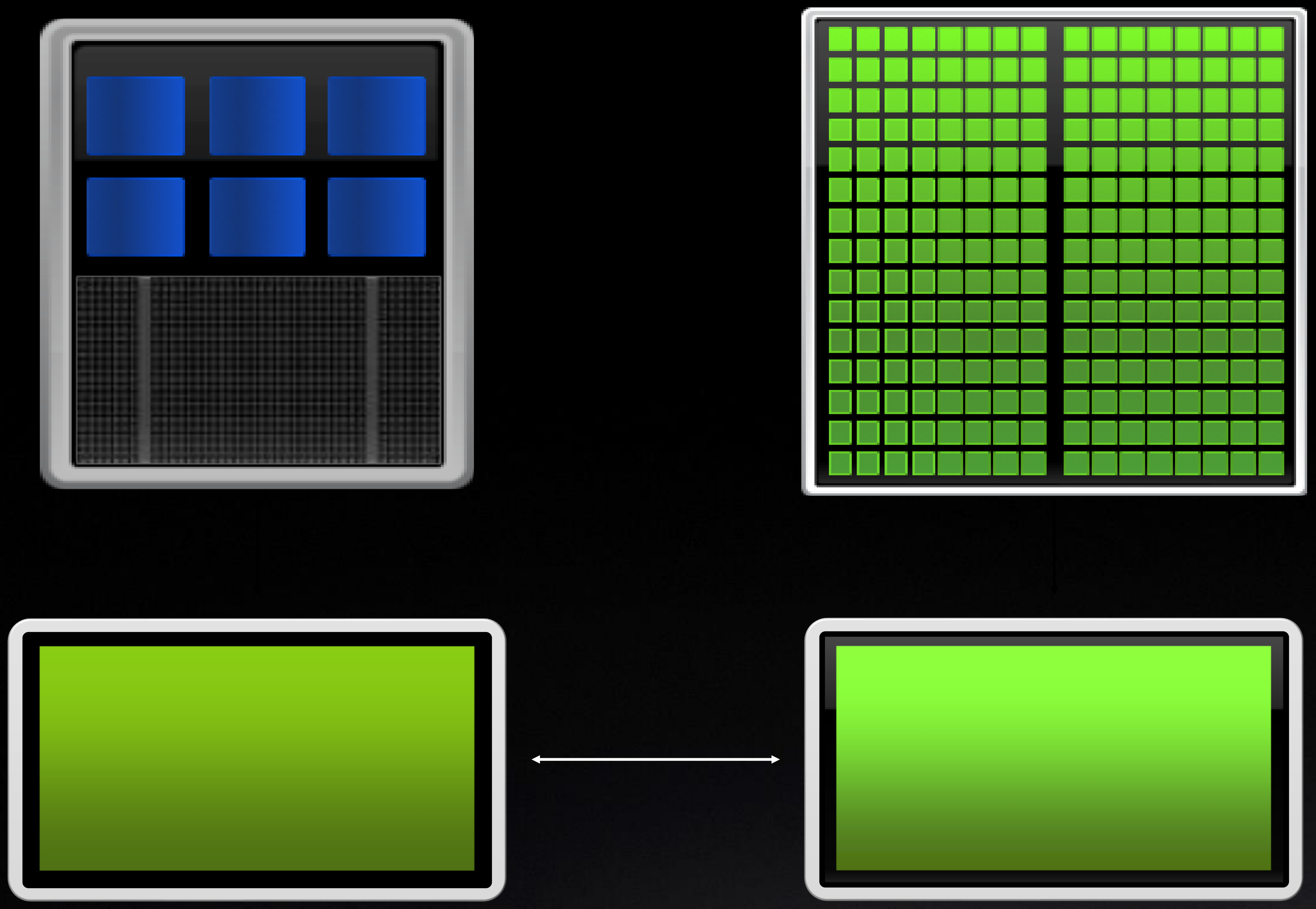
```
c = matmul(transpose(a),transpose(b))
c = matmul(transpose(a),reshape(b,shape=[k,n],order=[2,1]))
d = spread(a,dim=3,ncopies=nk)
d = spread(a,dim=1,ncopies=ni)
d = spread(a,dim=2,ncopies=nx)
d = alpha * abs(spread(a,dim=2,ncopies=nx))
d = alpha * spread(a,dim=2,ncopies=nx)
d = abs(spread(a,dim=2,ncopies=nx))
d = transpose(a)
d = alpha * transpose(a)
d = alpha * ceil(transpose(a))
d = alpha * conjg(transpose(a))
c = c + matmul(a,b)
c = c - matmul(a,b)
c = c + alpha * matmul(a,b)
d = alpha * matmul(a,b) + c
d = alpha * matmul(a,b) + beta * c
```

UNIFIED MEMORY

CUDA UNIFIED MEMORY

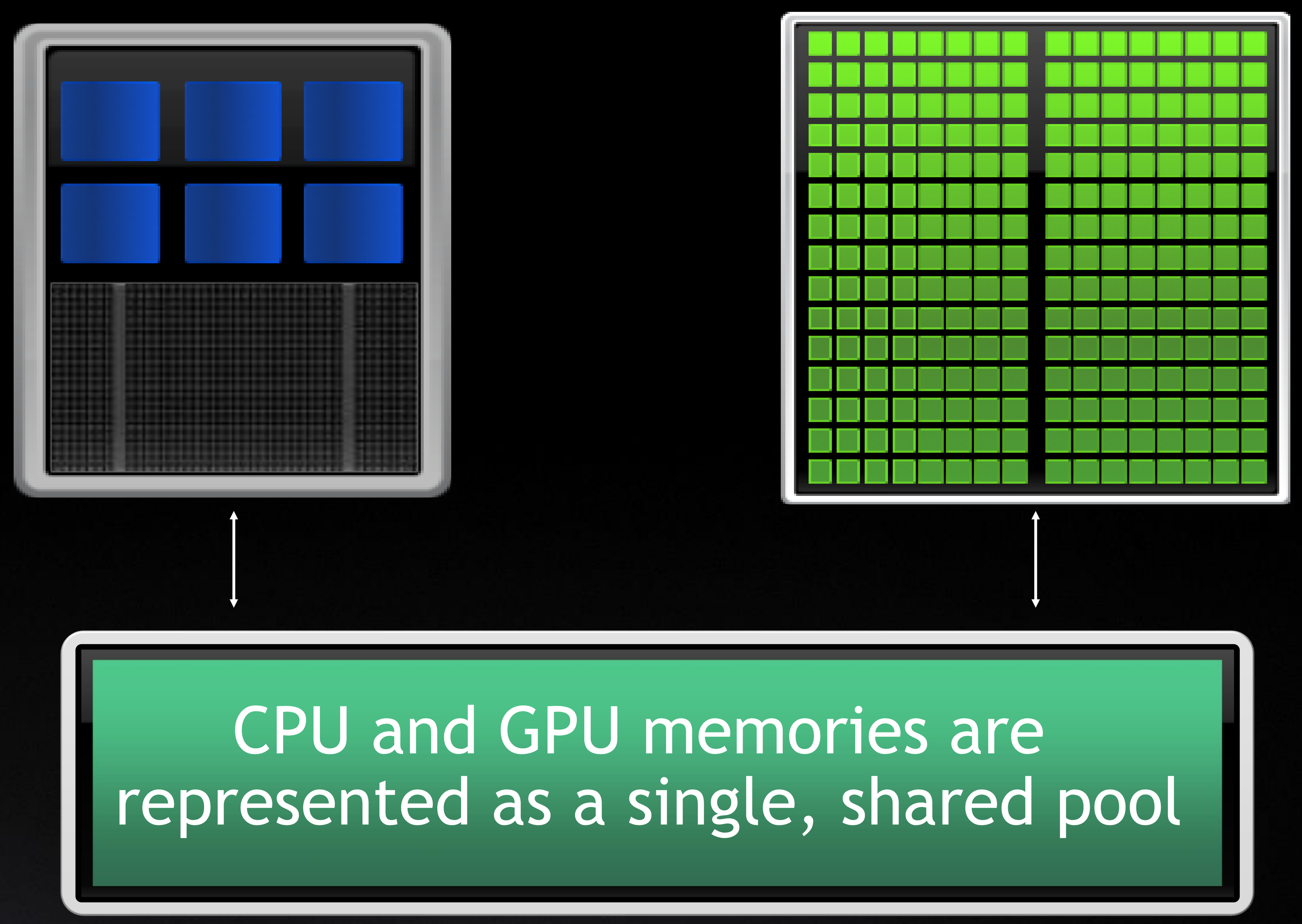
Simplified Developer Effort

Commonly referred to as *"managed memory."*



System Memory

GPU Memory

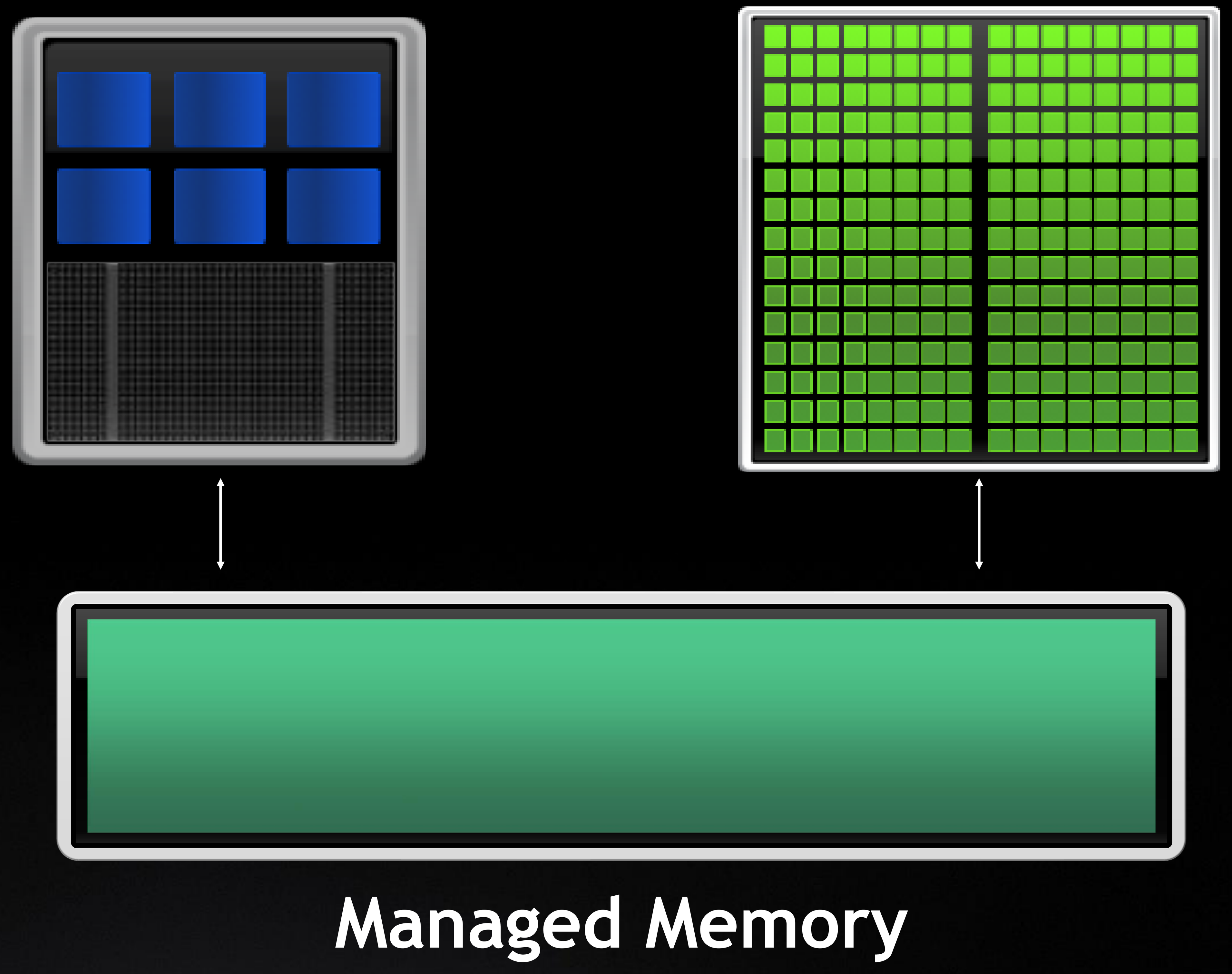


Managed Memory

MANAGED MEMORY

Limitations

- Disadvantages:
  - The programmer will almost always be able to get better performance by manually handling data transfers. Just Too Late
  - Memory allocation/deallocation takes longer with managed memory
- Advantages:
  - Handling explicit data transfers between the host and device (CPU and GPU) can be difficult
  - This allows the developer to concentrate on parallelism and think about data movement as an optimization
  - Supports oversubscription



CUDA MANAGED MEMORY

- OpenACC:

```
$ nvc++ –fast –ta=tesla:managed –Minfo=accel main.c
```

  - Enabled using –ta=tesla:managed

- std::par

```
nvc++ -stdpar=gpu program.cpp -o program
```

  - *All* allocations use managed memory

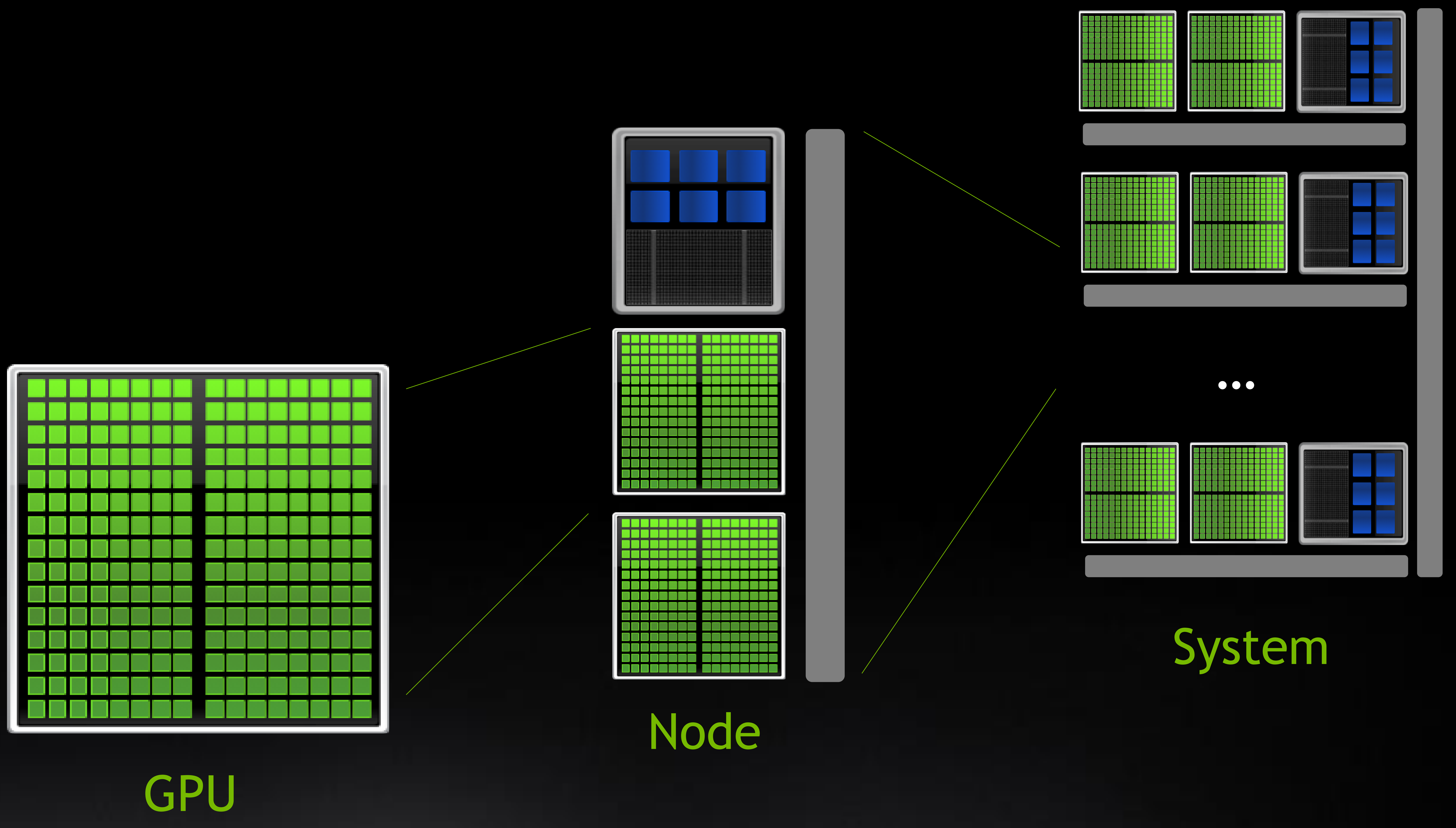
- OpenMP:
  - Current Beta release does not support Unified memory. Need explicitly use *target map* directive to copy data

AT SCALE

OUTSIDE OF THE GPU

Accelerating at all scales

- PCIe3/4
- MPS (GROMACS blog)
- NVLink
- NVSwitch
- GPUDirect
  - Peer to peer
  - RDMA
  - Storage
- DPU - Bluefield



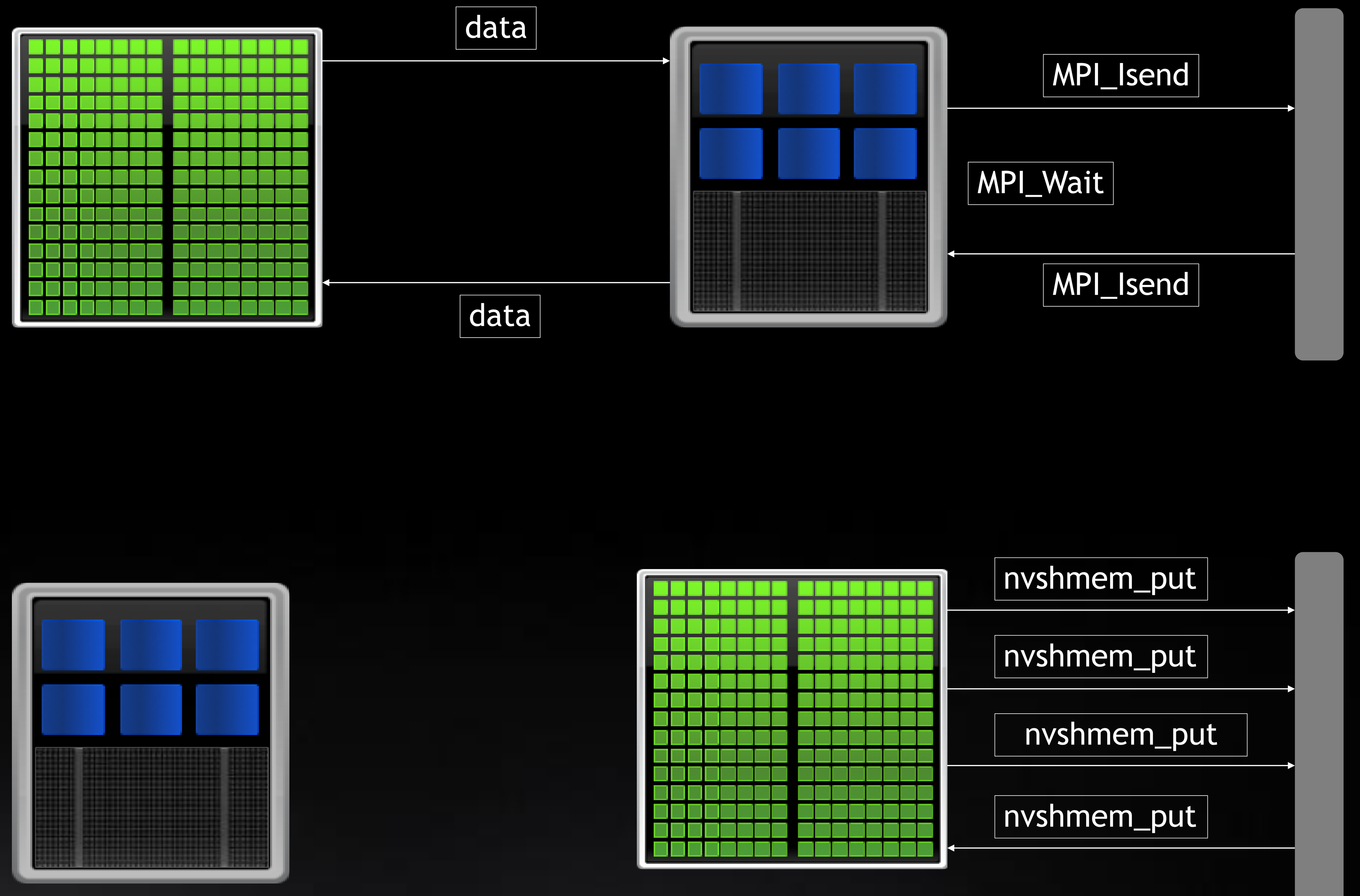
INTRODUCING NVSHMEM

GPU Optimized OpenSHMEM

- Initiate from CPU or GPU
- Initiate from within CUDA kernel
- Issue onto a CUDA stream
- Interoperable with MPI & OpenSHMEM

Pre-release Impact

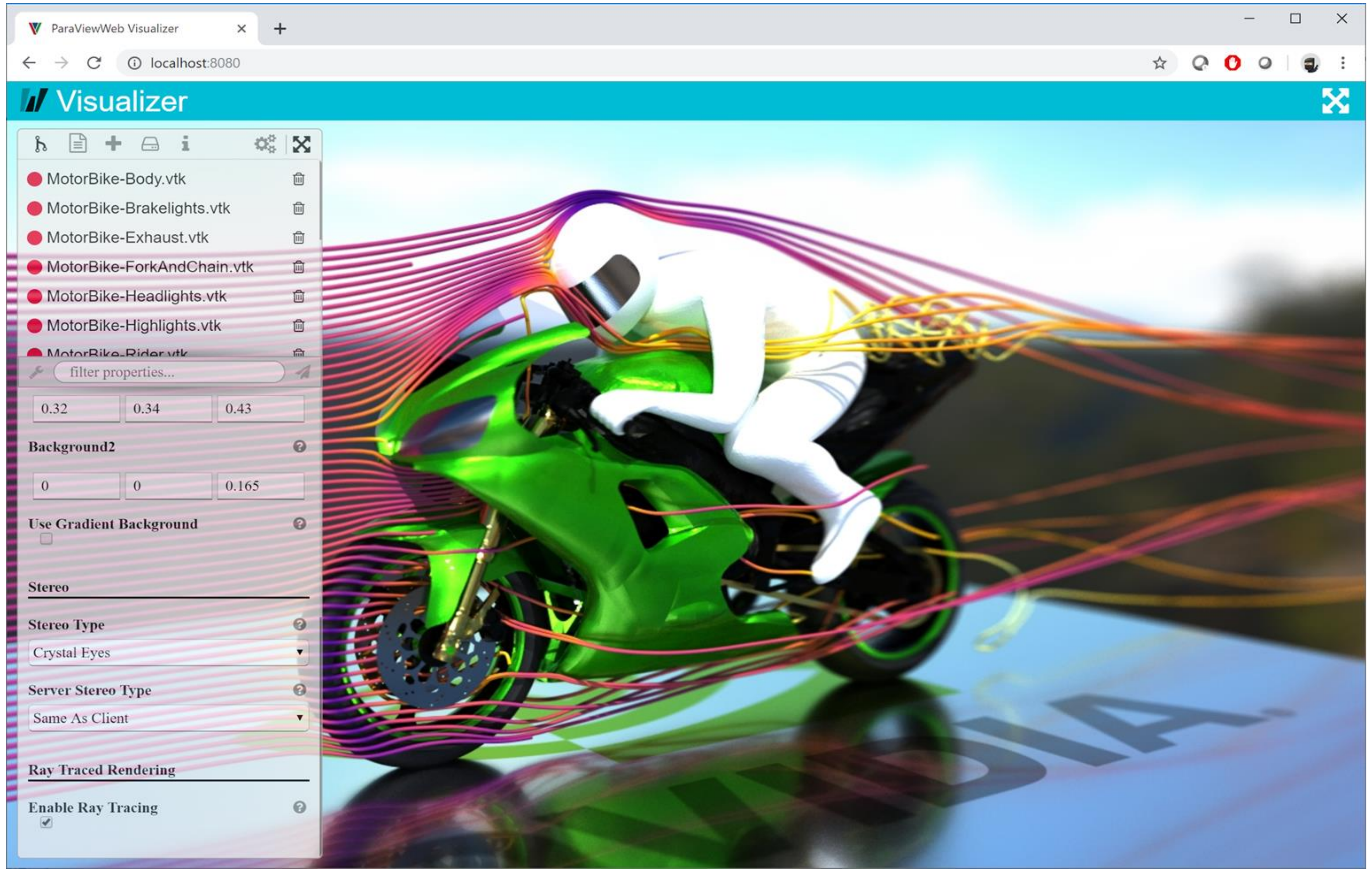
- LBANN, Kokkos/CGSolve, QUDA



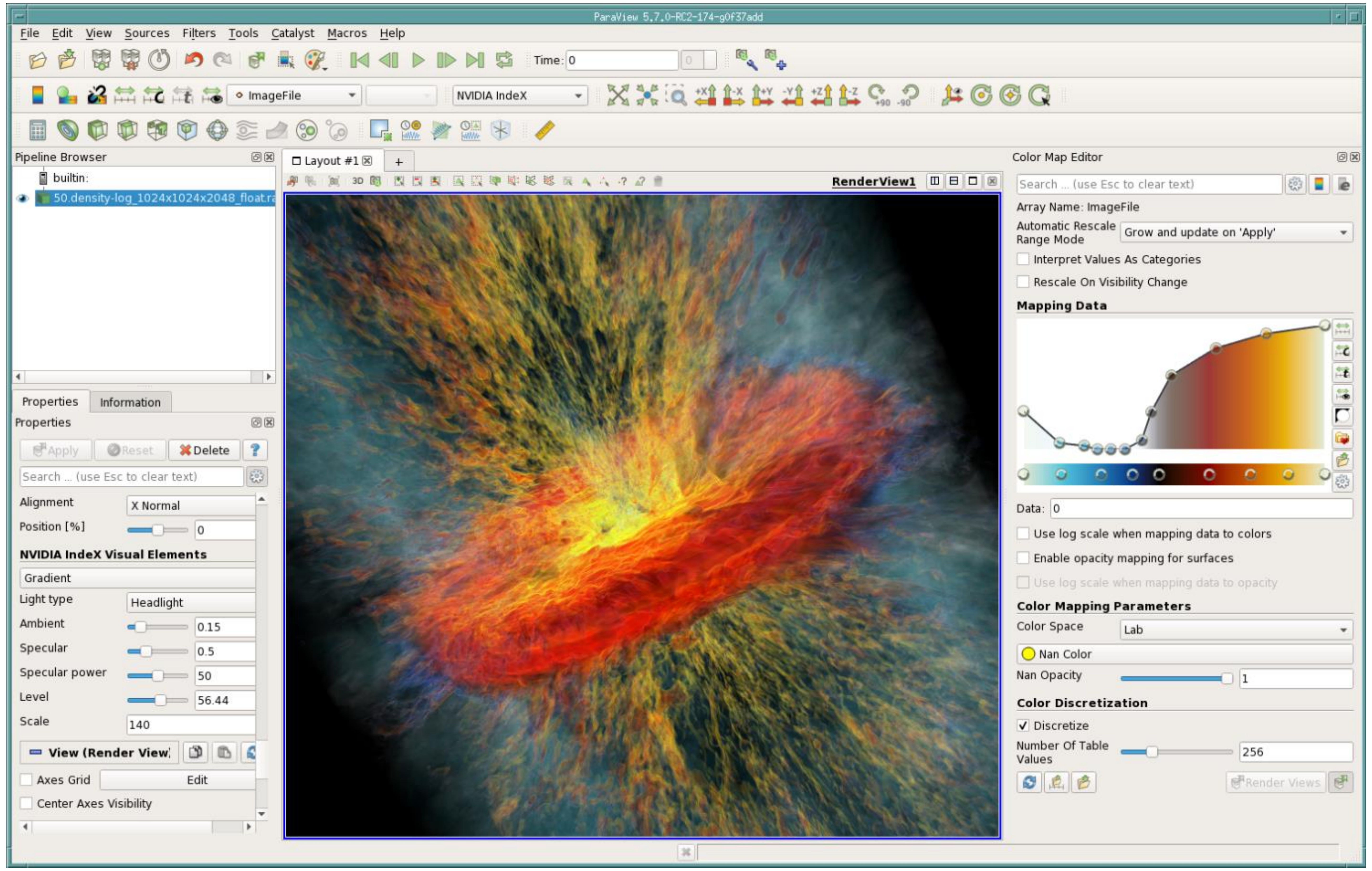
VISUALISATION

STANDARD WORKFLOW, REINVENTED

NVIDIA Rendering Technology in Open Source Vis Tools



OptiX Ray Tracing Backend



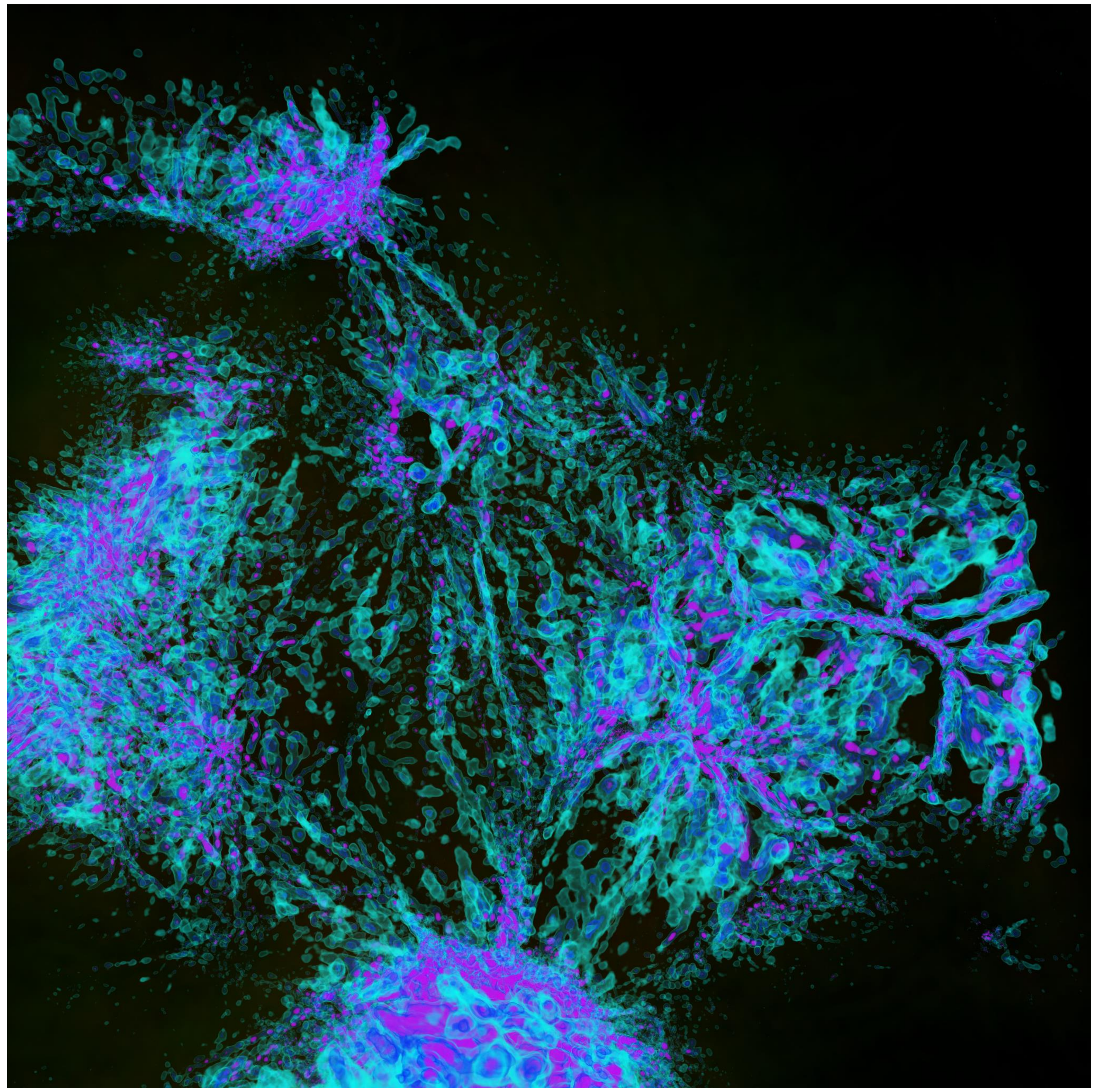
IndeX Volume Rendering Plugin

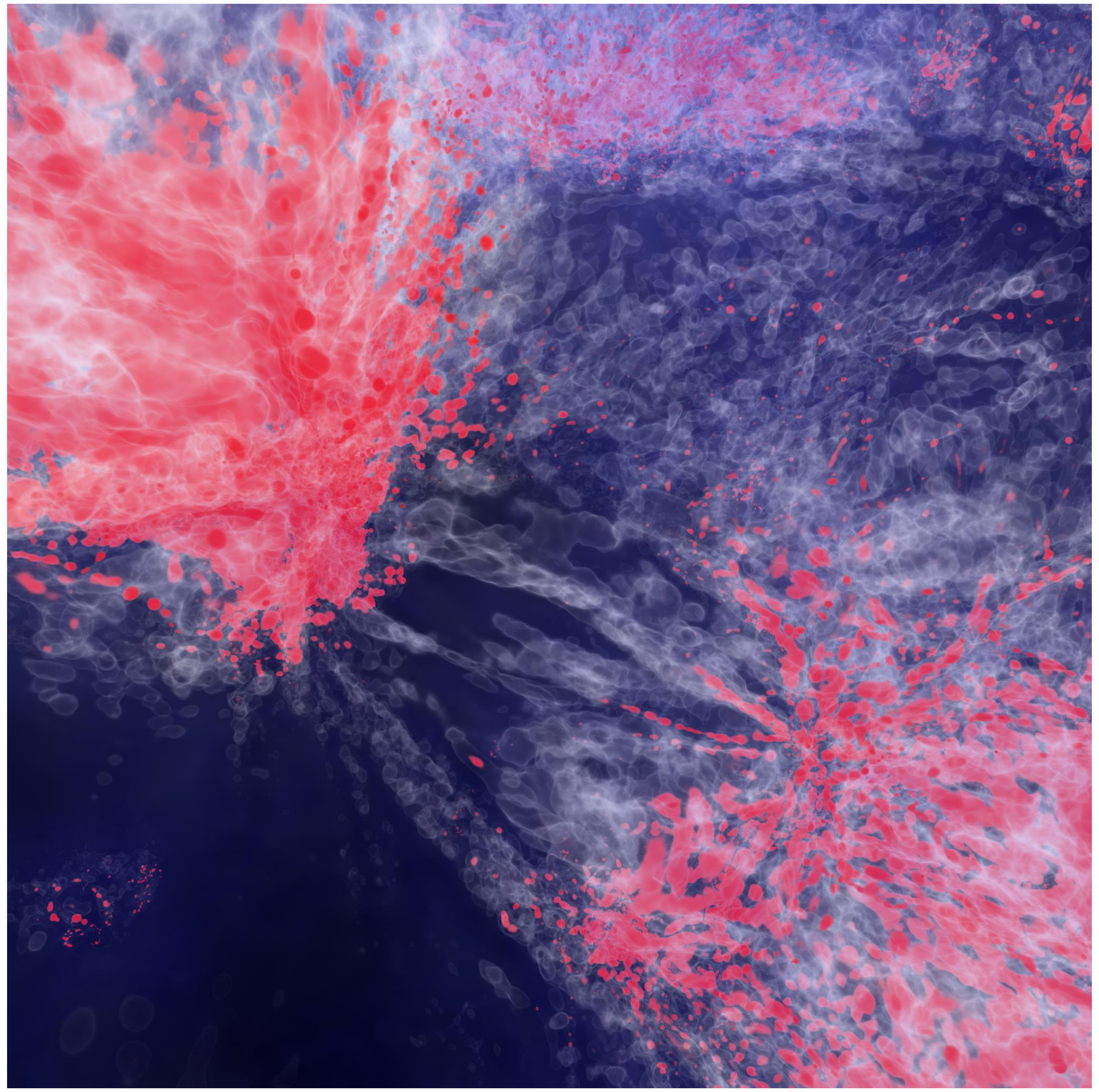
https://www.paraview.org/download/

SCALABLE POINT-CLOUD RENDERING

RBF Point Cloud rendering in NVIDIA IndeX

- Accumulation of RBF contributions along rays
- Multi-GPU, multi-node support
- Leverages RTCores
- V100->RTX8000: 70% BW, 2x speedup
- 17M ptcls, 8 GPUs, 10–20 fps
- Release upcoming





Small scale Cholla simulation, 17M ptcls, 8GPUs.
Data courtesy of B. Robertson, E Schneider

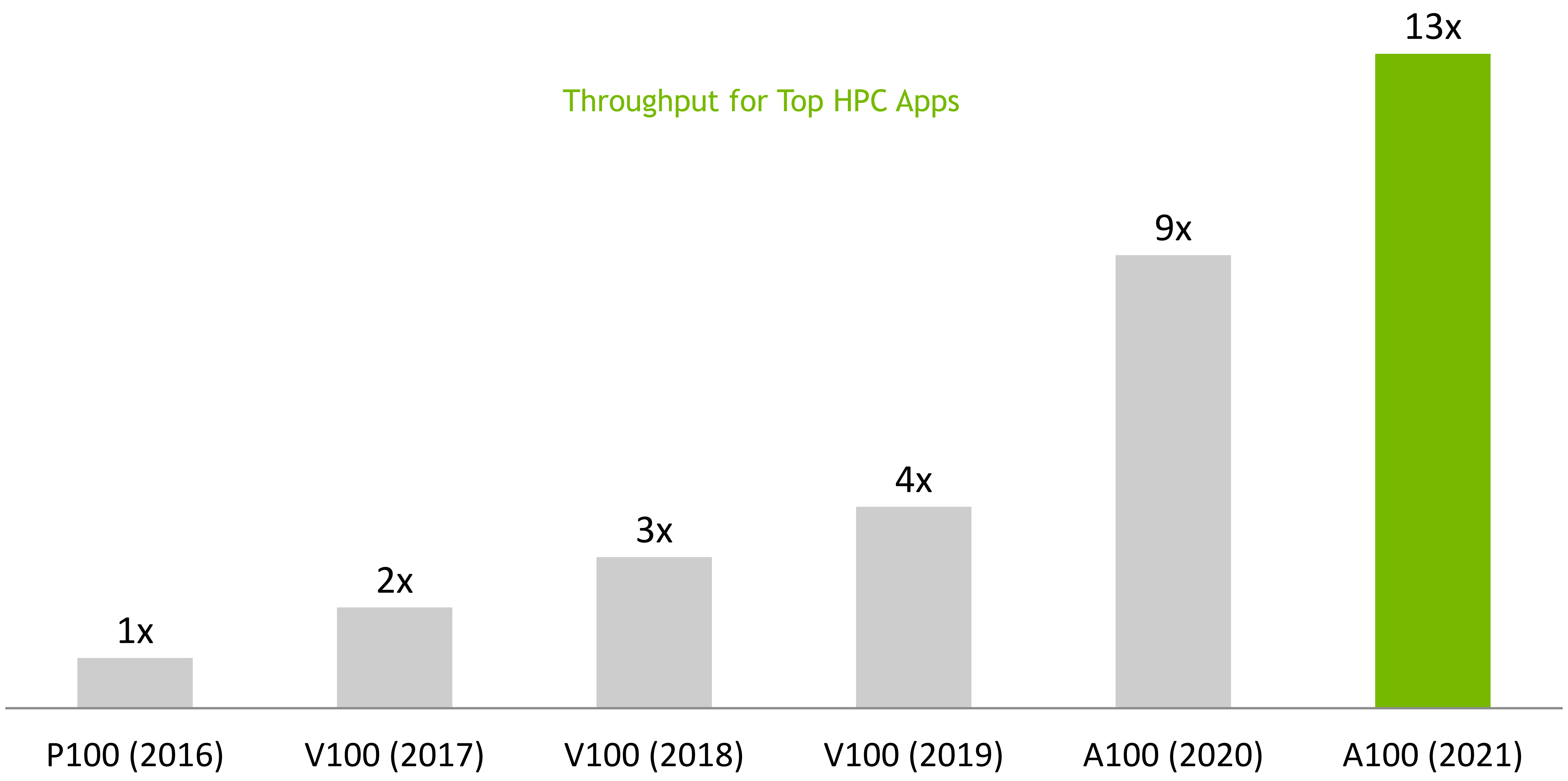
TORNADO VISUALIZATION

IndeX, GPUDirect Storage, Iray, OmniVerse

https://www.youtube.com/watch?v=SonfENaSesw

FINAL WORD

KEEP APPS, LIBRARIES AND FRAMEWORKS UP TO DATE



Geometric mean of application speedups vs. P100: Benchmark application: Amber [PME-Cellulose_NVE], Chroma [szscl21_24_128], GROMACS [ADH Dodec], MILC [Apex Medium], NAMD [stmv_nve_cuda], PyTorch (BERT-Large Fine Tuner], Quantum Espresso [AUSURF112-jR]; Random Forest FP32 [make_blobs (160000 x 64 : 10)], TensorFlow [ResNet-50], VASP 6 [Si Huge] | GPU node with dual-socket CPUs with 4x NVIDIA P100, V100, or A100 GPUs.

RESOURCES

RELATED TALKS AT GTC

All times GMT. GTC runs March 21st-24th , register for free

- Tue 22nd 5pm - A Deep Dive into the Latest HPC software: link
- Mon 21st 6pm – Connect with the Experts: Best Practices for Fortran on GPUs: link
- Thu 24th 8pm – From Directives to DO CONCURRENT: A Case Study in Standard Parallelism
- Wed 23rd 2pm - Recent Developments in NVIDIA Math Libraries: link
- Mon 21st 9pm – Connect with the Experts: NVIDIA Math Libraries: link
- Tue 22nd 9pm – C++ Standard Parallelism: link
- Mon 21st 4pm – No More Porting: Coding for GPUs with Standard C++, Fortran and Python: link
- Tue 22nd 8pm - Multi-GPU programming with MPI: link
- Thu 24th 1pm - NVSHMEM: CUDA-Integrated Communication for NVIDIA GPUs: link
- Thu 24th 1pm - Optimizing Communication with Nsight Systems Network Profiling: link
- Wed 23rd 7pm – Latest on NVIDIA Magnum IO GPUDirect Technologies: link
- Thu 24th 8pm – Accelerating Storage IO to GPUs with Magnum IO: link
- Thu 24th 6pm – Visualizing the World's Most Violent Tornadoes using NVIDIA IndeX in Omniverse: link
- Tue 22nd 5pm – Connect with the Experts: CUDA Memory Management: link
- Mon 21st 10pm – Connect with the Experts: Directive-based GPU Programming with OpenACC: link
- The 24th 4pm – Optimizing GPU Utilization: Understanding MIG and MPS: link
- … session catalogue: https://www.nvidia.com/gtc/session-catalog/

NVIDIA

NGC: GPU-OPTIMIZED SOFTWARE HUB

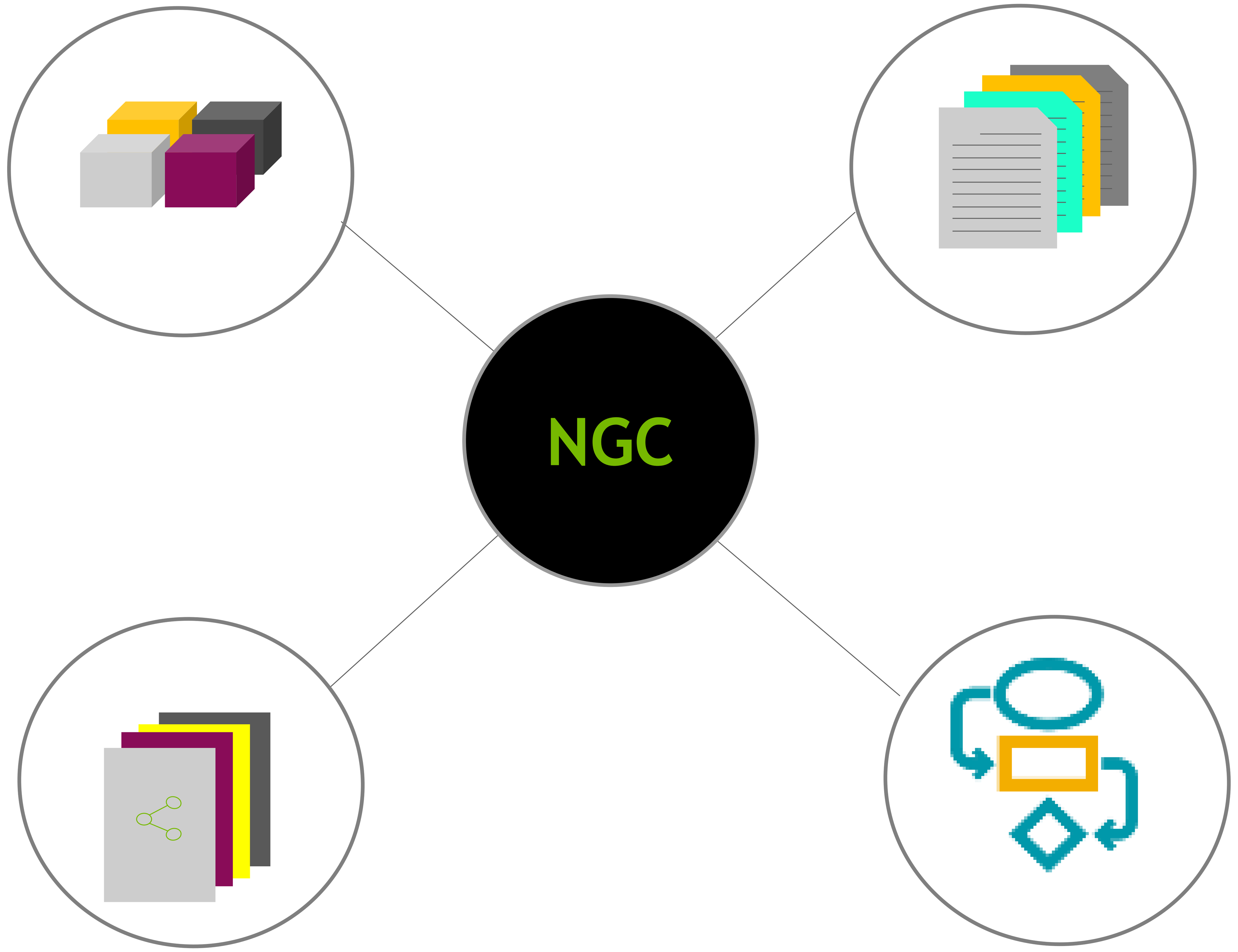
Simplifying DL, ML and HPC Workflows

50+ Containers

DL, ML, HPC

15+ Model Training Scripts

NLP, Image Classification, Object Detection and more



60 Pre-trained Models

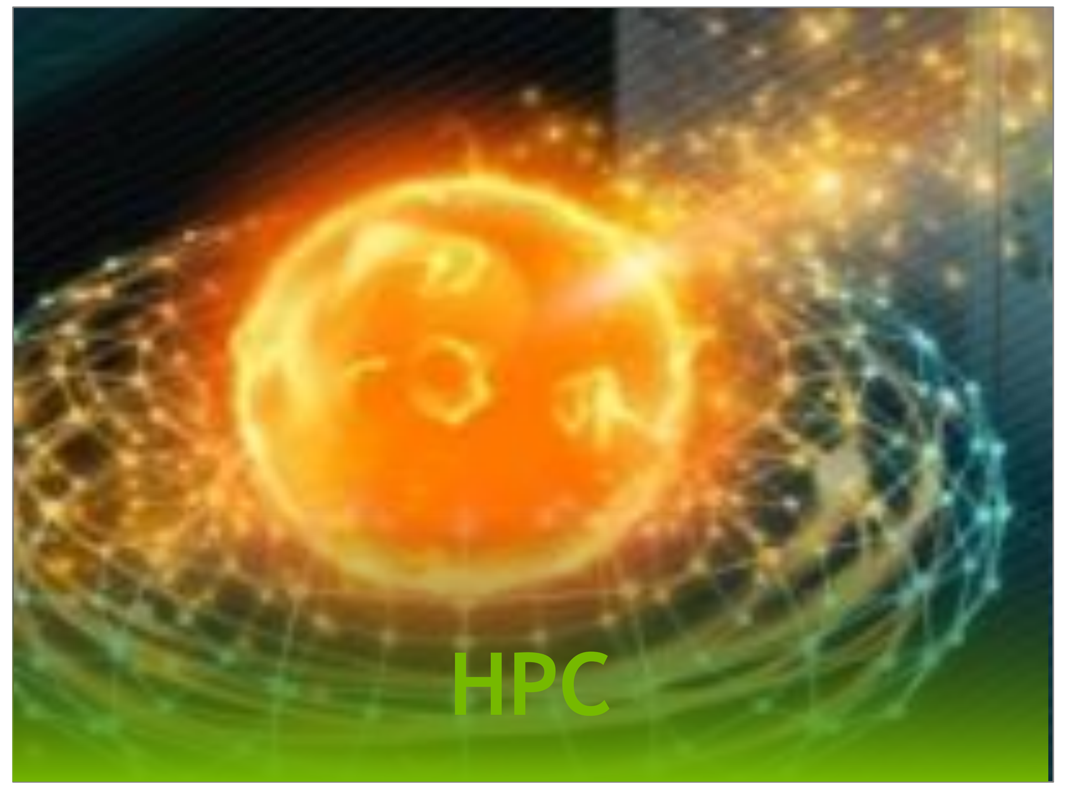
NLP, Image Classification, Object Detection and more

Workflows

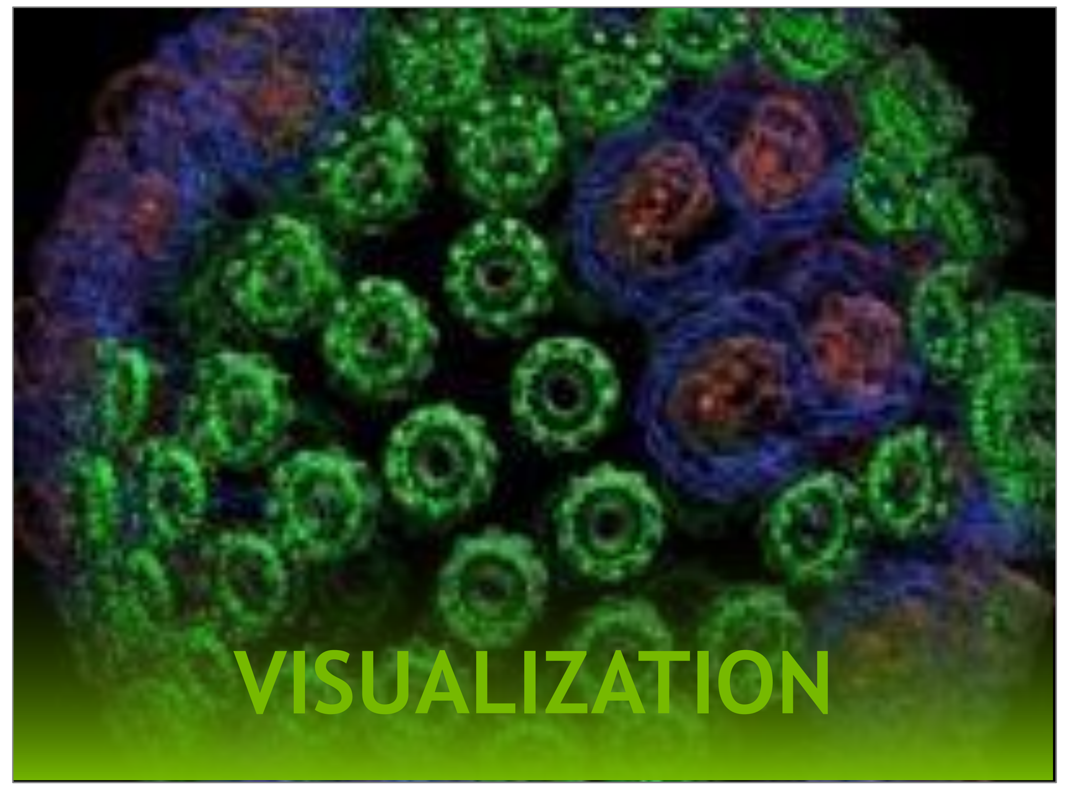
Medical Imaging, Intelligent Video Analytics

TensorFlow | PyTorch | more

RAPIDS | H2O | more



NAMD | GROMACS | more



ParaView | IndeX | more

NVIDIA

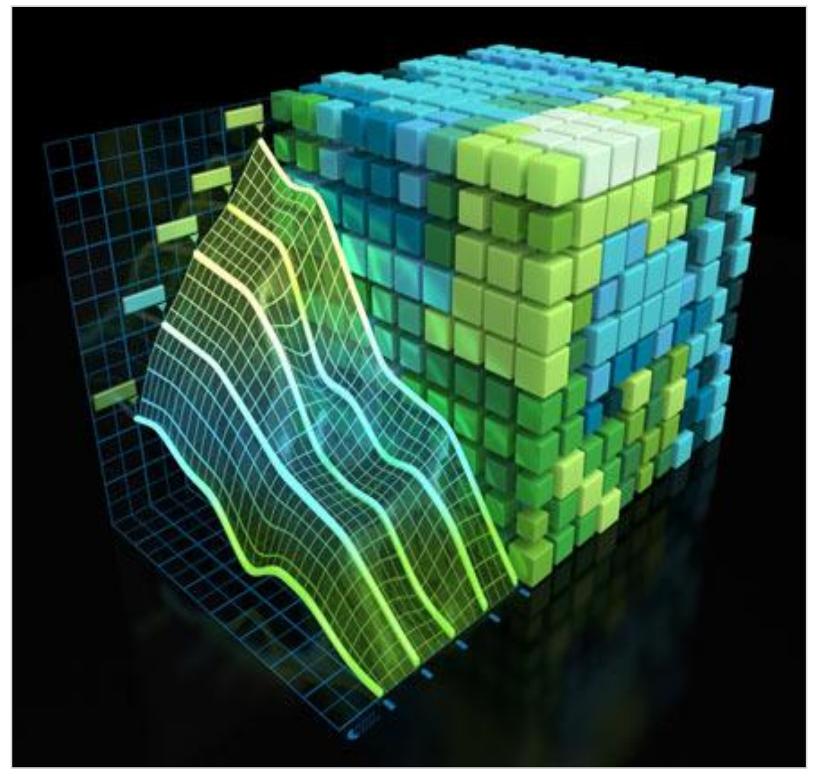
DEEP LEARNING INSTITUTE (DLI)

Hands-on, self-paced and instructor-led training in deep learning and accelerated computing

Request onsite instructor-led workshops at your organization:
www.nvidia.com/requestdli

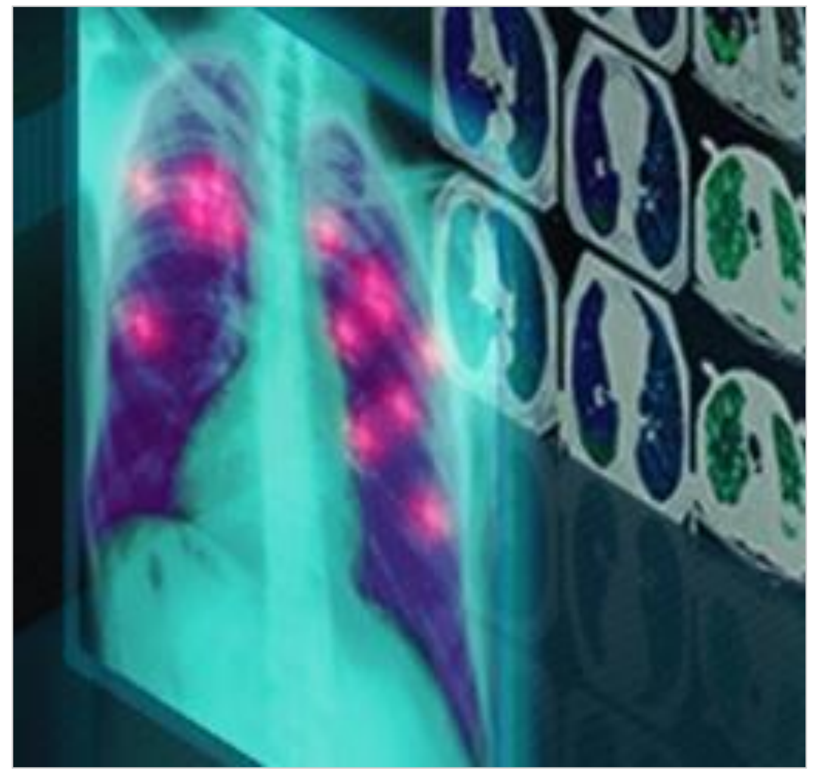
Take self-paced courses online:
www.nvidia.com/dlilabs

Download the course catalog, view upcoming workshops, and learn about the University Ambassador Program:
www.nvidia.com/dli

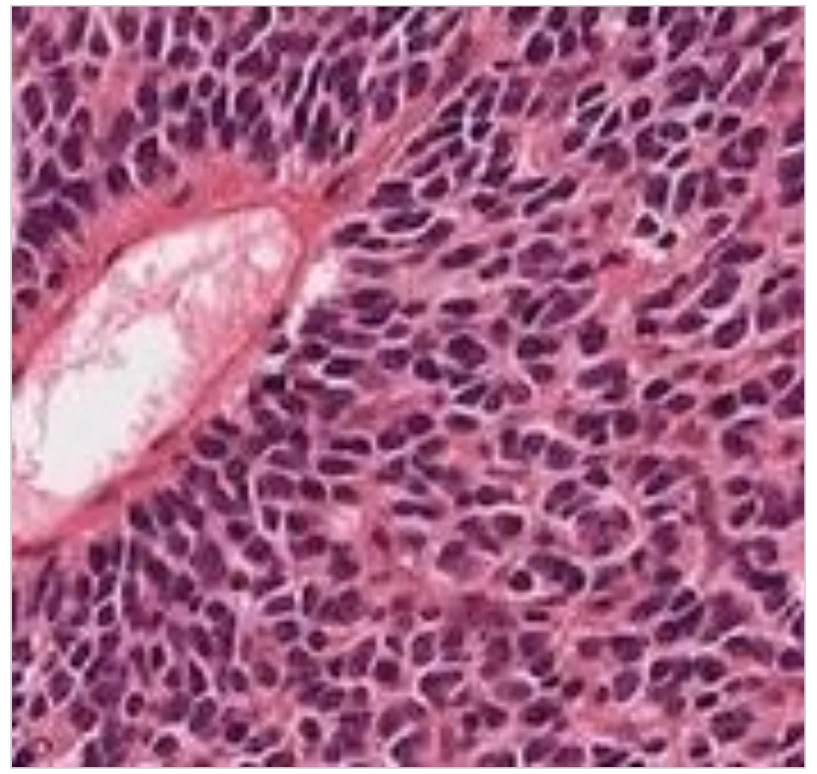


Accel. Computing Fundamentals

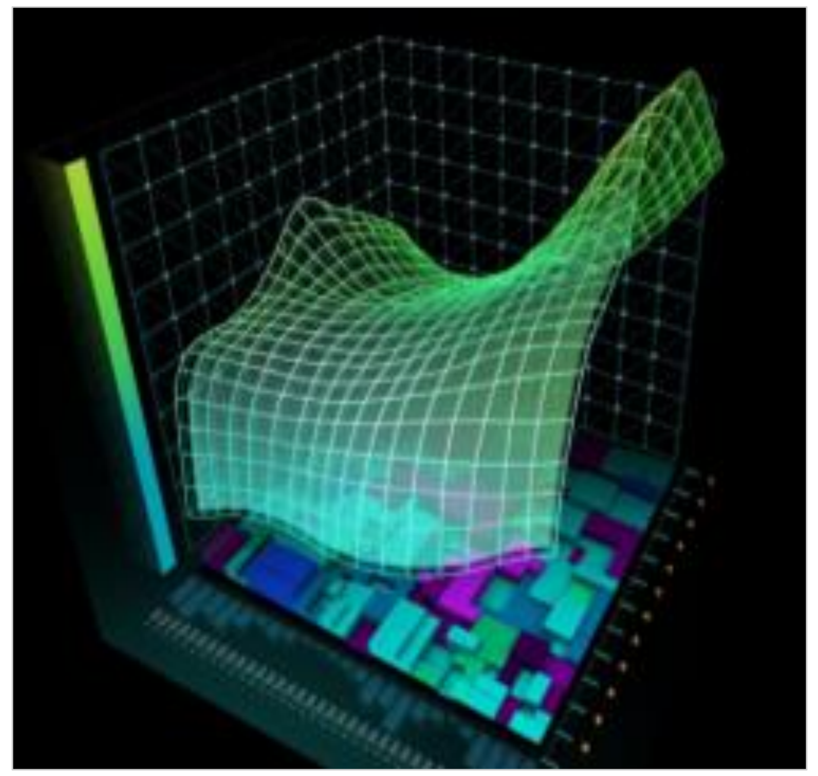
Autonomous Vehicles



Medical Image Analysis



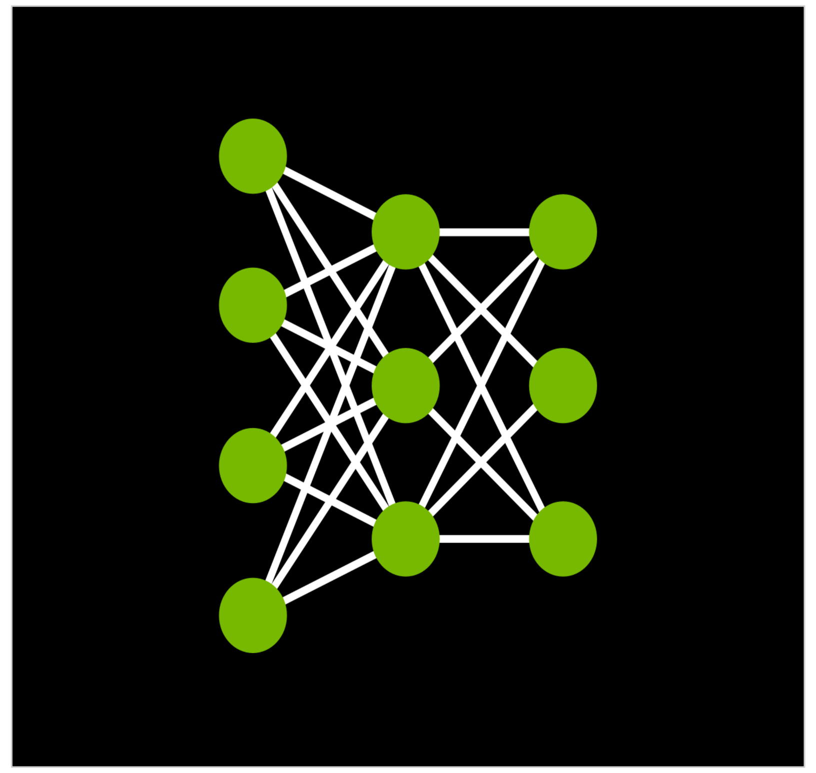
Genomics



Finance

Digital Content Creation

Game Development



Deep Learning Fundamentals

More industry-specific training coming soon…

PUBLIC INSTRUCTOR-LED WORKSHOP SCHEDULE

All workshops are offered remotely in a virtual classroom

Fundamentals of Accelerated Computing with CUDA C/C++
Mon, May 23, 9:00 a.m. to 5:00 p.m. CEST (EMEA)

Scaling CUDA C++ Applications to Multiple Nodes
Tue, May 24, 9:00 a.m. to 5:00 p.m. CEST (EMEA)

Accelerating Data Engineering Pipelines
Tue, Jun 14, 9:00 a.m. to 5:00 p.m. CEST (EMEA)

To register visit courses.nvidia.com/public

DEVELOPER ENGAGEMENT PLATFORMS

Information, downloads, special programs, code samples, and bug submission	developer.nvidia.com
Container repository for tools, deep learning frameworks, HPC applications, vizualisation tools	ngc.nvidia.com
Insights & help from other developers and NVIDIA technical staff	devtalk.nvidia.com
Technical documentation	docs.nvidia.com
Deep Learning Institute: workshops & self-paced courses	courses.nvidia.com
In depth technical how to blogs	devblogs.nvidia.com
Developer focused news and articles	news.developer.nvidia.com
Webinars	nvidia.com/webinar-portal

RESOURCES AVAILABLE TO ACADEMICS

Developer Teaching Kits: which include free access to online training for students but they have to be requested by a lecturer/professor.

Academic Workshops:
The NVIDIA website lists free academic workshops that our Ambassadors are giving around the world that you can go and attend

Bootcamps:
~ 2 day tailored training events, typically for a target group e.g. OpenACC, AI for Science

Hackathons:
In-depth events with access to expert mentors

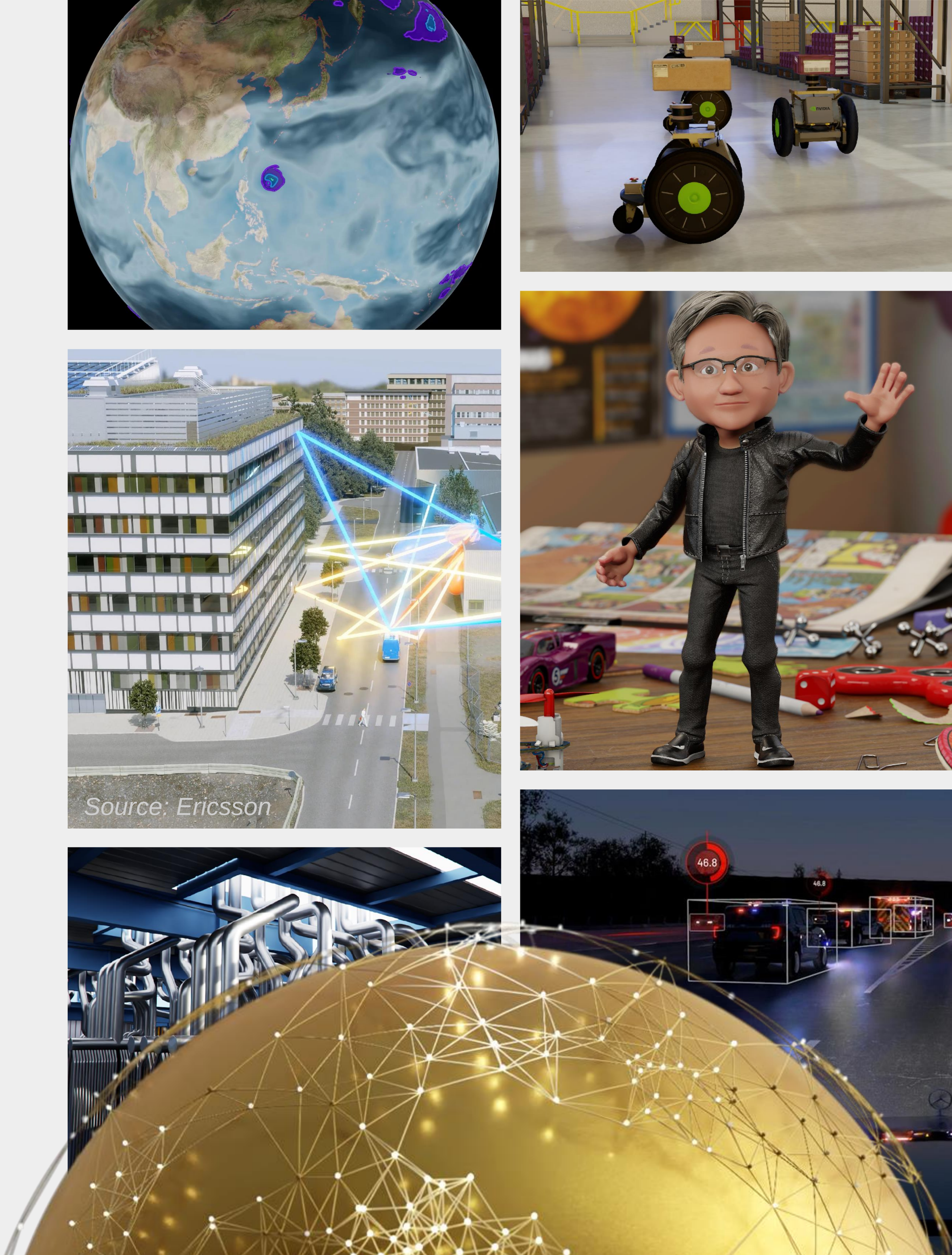
NVIDIA GTC

YOUR MOST BRILLIANT WORK STARTS HERE

The Developer Conference for the Era of AI

NVIDIA GTC is more than a game-changing AI developer conference. It's a global community committed to decoding the world's greatest challenges, transforming every major industry workflow, and exploring tomorrow's next big ideas—together. Join us this March and discover how to accelerate your life's work.

MARCH 21—24, 2022 | www.nvidia.com/GTC



Paul Graham | pgraham@nvidia.com