# AI as a Detector: Real Time AI Technosignature Search

Adam Thompson | Head of Product – Edge Supercomputing | **NVIDIA**

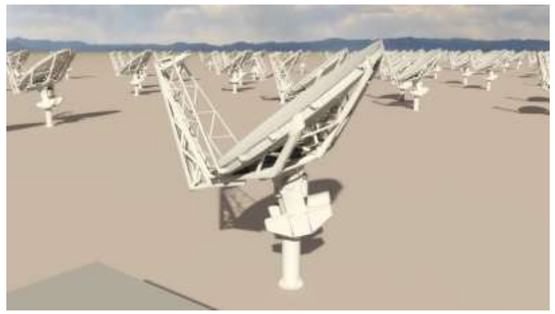
# Next Generation Instruments are Producing Overwhelming Amounts of Data

## Complexity of Experiments is Booming and Human Insight is Now the Bottleneck

### Radio Astronomy

ngVLA – 244 Dishes

**100 Petabytes per Year**



SKA – 200 Dishes

**1 Exabyte per Year**

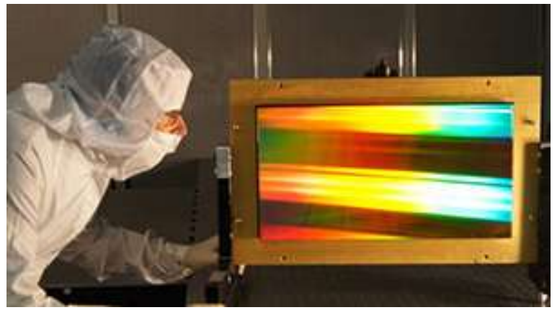
### Particle Physics

High Luminosity LHC

**100x Data than Higgs Boson Discovery**

Advanced Laser Systems

**100x Increase in Repetition Rate**



### Light Sources

APS-U – >60 beamlines

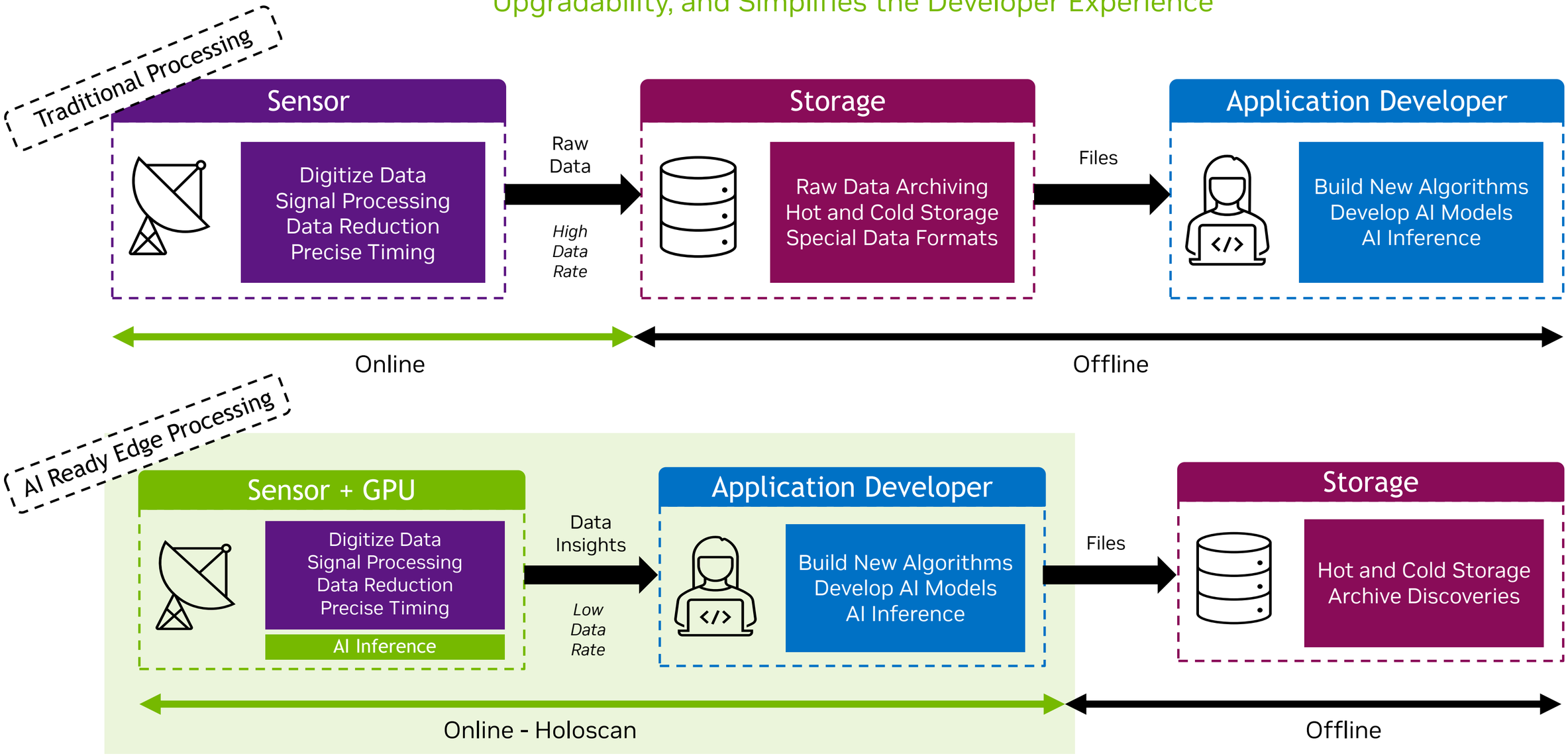
**100-200 Petabytes per Year**

Free Electron Laser LCLS-II

**1 MHz Rep-Rate Upgrade**

# Resolving the Data Deluge with Edge Supercomputing

## Online AI Processing Enables Autonomous Experiments, Reduces Infrastructure Costs, Improves Upgradability, and Simplifies the Developer Experience



# Edge Supercomputing

AI Training | Simulation | Real Time Deployment

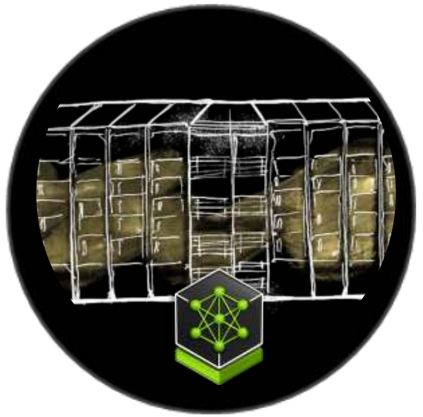
**Edge**
Real time, high data rate physical measurements

*Interactive Digital Twin*

**Simulation**
Digital twin and advanced simulation for experiment steering and configuration

*Continual Learning*

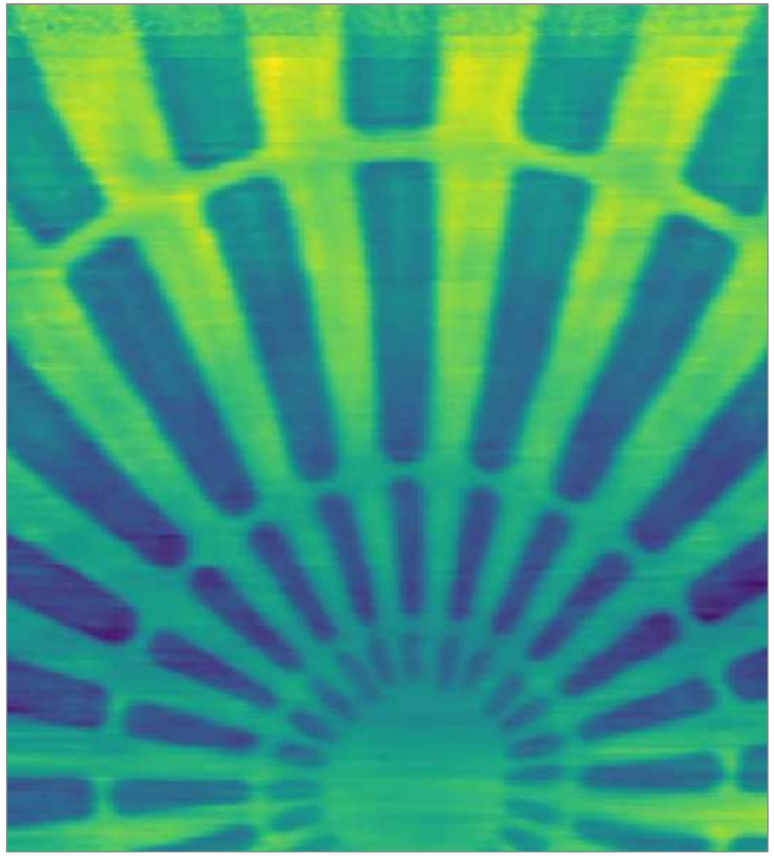
*Edge Informed Synthetic Data*



**AI Factory**
Supercomputers for training AI models at massive scale

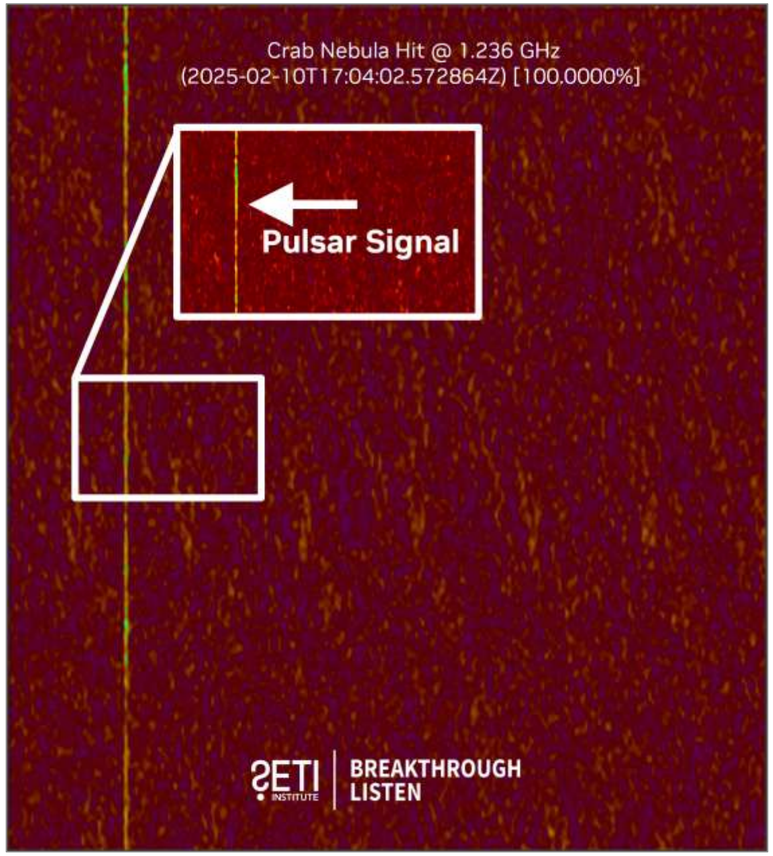
# Real Time Data Analysis at the Edge

AI, Combined With Streaming Data and Unprecedented Rates, Welcomes New Scientific Discoveries



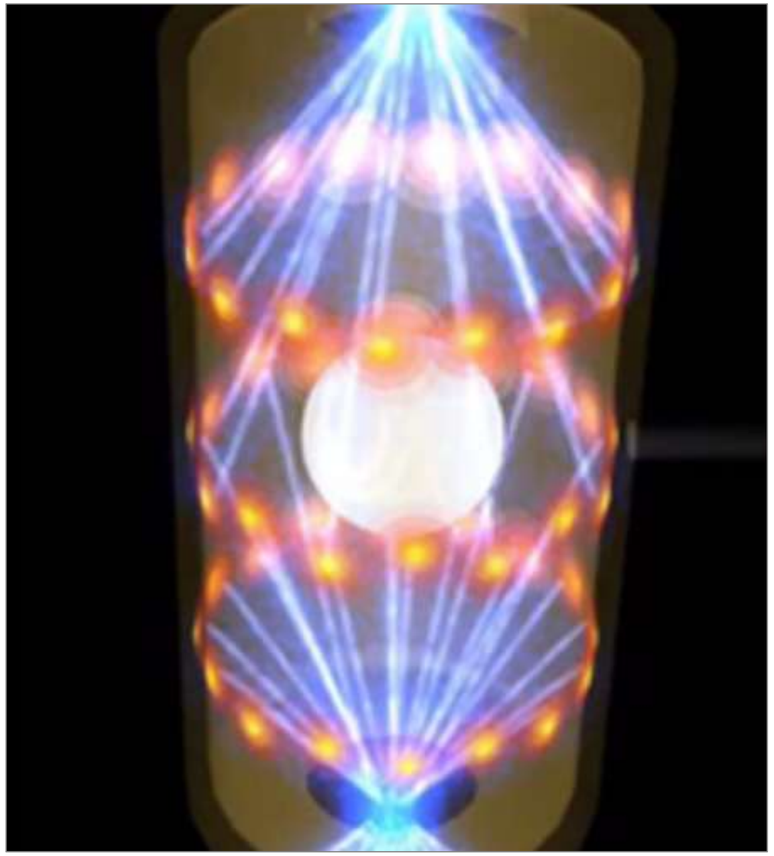
## Nanoscale Imaging of Materials

Real Time Ptychography with NSLS-II and DECTRIS




## AI Search for Pulsars and Technosignatures

AI as a Detector with the SETI Institute and the Allen Telescope Array



## Self Driving Experiments

AI Guided Fusion Research with Lawrence Livermore National Laboratory

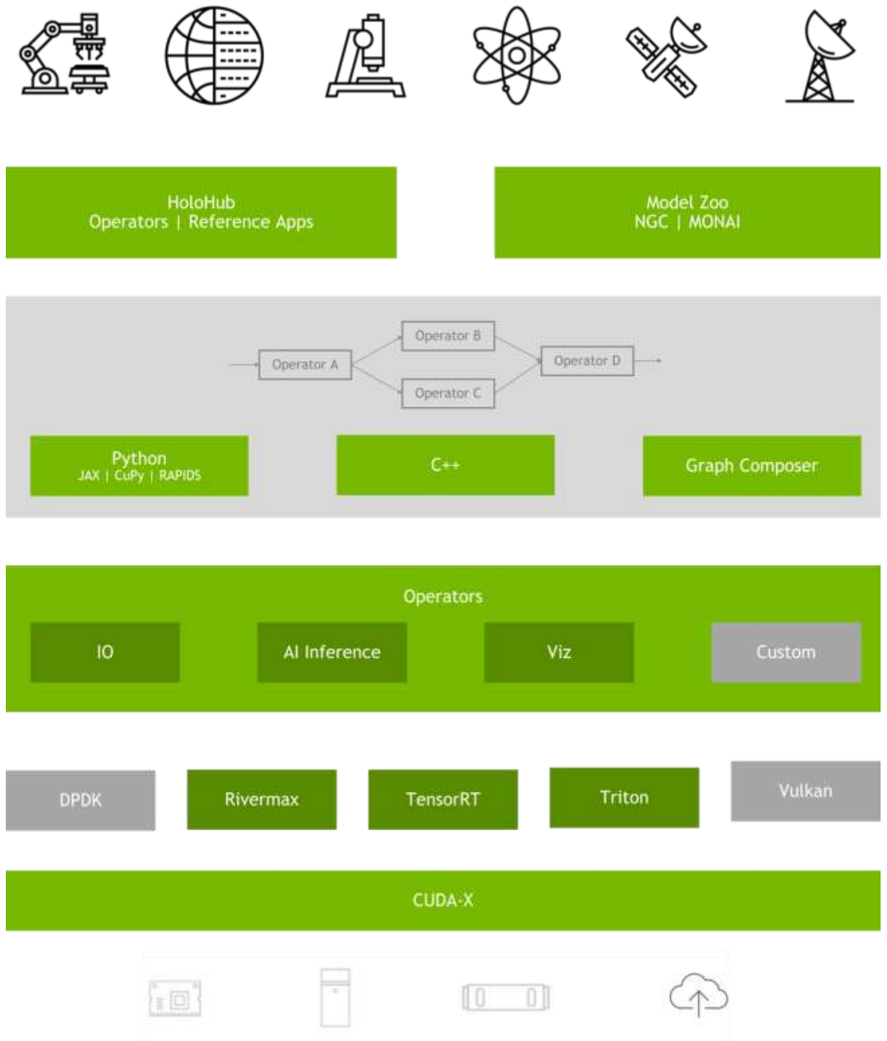
# Holoscan

# AI-Enabled, Real-Time Sensor Processing Platform

# NVIDIA Holoscan

## SDK for Building AI-Enabled Sensor Processing Applications

GitHub




## Features

- C++ and Python APIs for domain agnostic sensor data processing workflows
- Multi-Node and Multi-GPU support with advanced pipeline scheduling options and network-aware data movement
- AI Inference with pluggable backends such as ONNX, Torchscript, and TensorRT
- Scalable from Jetson Orin Nano (ARM + GPU) to Grace Hopper
- Apache 2 Licensed and Available on GitHub

## Benefits

- Simplifies sensor I/O to GPU
- Simplifies the performant deployment of an AI model in a streaming pipeline
- Provides customizable, reusable, and flexible components to build and deploy GPU-accelerated algorithms
- Scale workloads with Holoscan's distributed computing features
- Deploy to the Cloud with Holoscan App Packager and Kubernetes

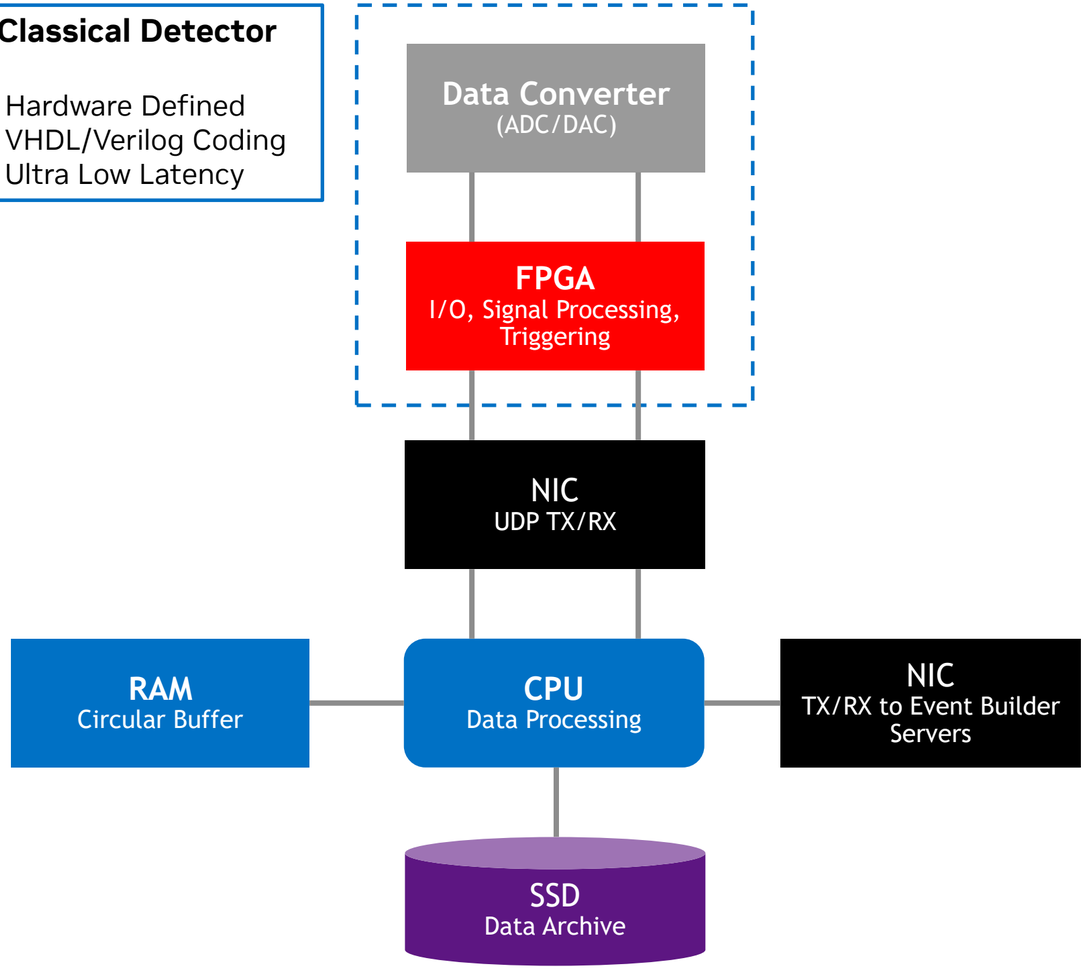
NVIDIA

# AI Enabled and Science Programmable Data Acquisition (DAQ)

## Traditional DAQ Architecture

**Classical Detector**

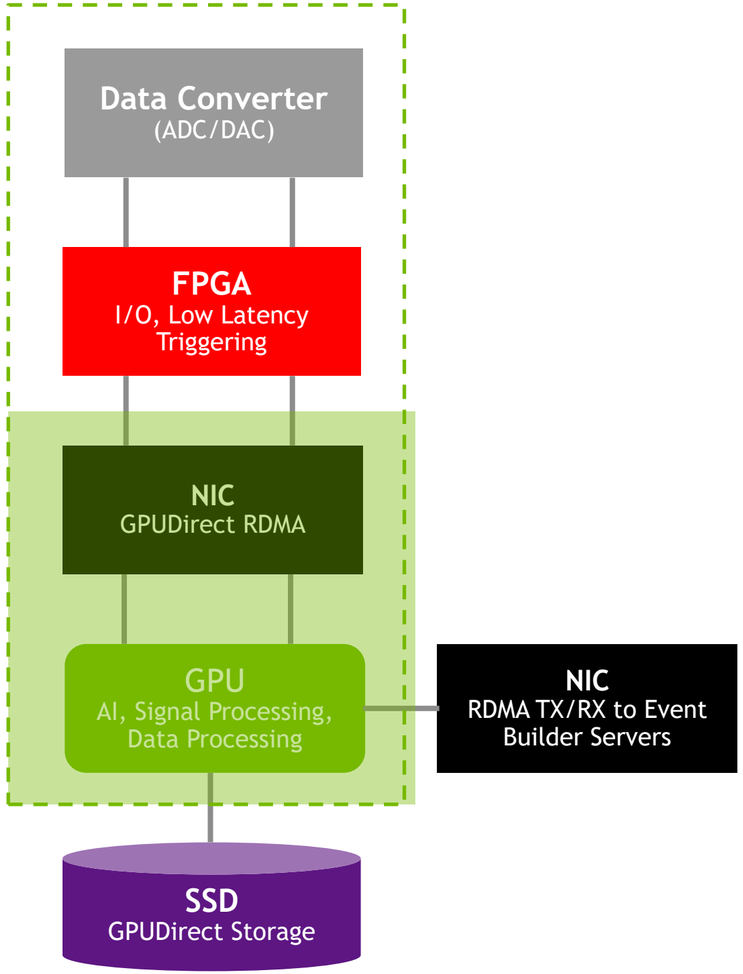
- Hardware Defined
- VHDL/Verilog Coding
- Ultra Low Latency




## AI Ready DAQ Architecture

**GPU-Centric Detector**

- Software Defined
- Python / C++ Coding
- AI in the Loop
- FPGA for Data Conversion and Low Latency Signal Processing
- Data direct to GPU with GPUDirect or RDMA



# Holoscan Advanced Network Operator

## Simplification of Moving Data To and From GPU at Line Rate

### Advanced Network Operator

Focus on **Performance**: Any Ethernet Packet (UDP, VITA-49, etc) to and from GPU at Line Rate

Bypasses Linux kernel for access directly to NIC DMA buffers

Requires a Mellanox/NVIDIA Network Interface Card (NIC)

Zero-copy interface from NIC into user buffers or directly to GPU

Supports multiple backends (DOCA DPDK, Rivermax, GPUNetIO, RoCEv2) with YAML configuration

Python bindings are in progress

**Learn more about the ANO on Holohub**

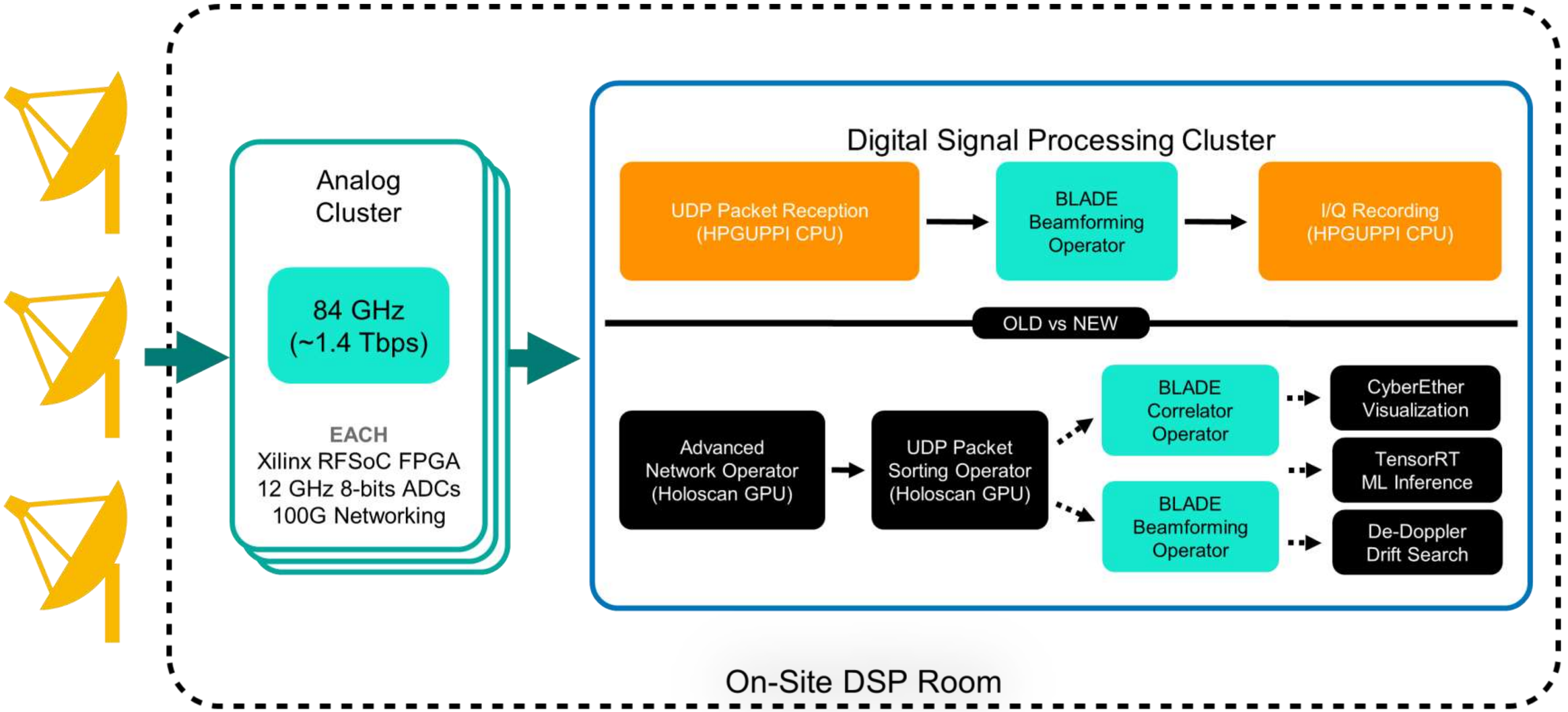

### YAML Configuration - Header Data Split

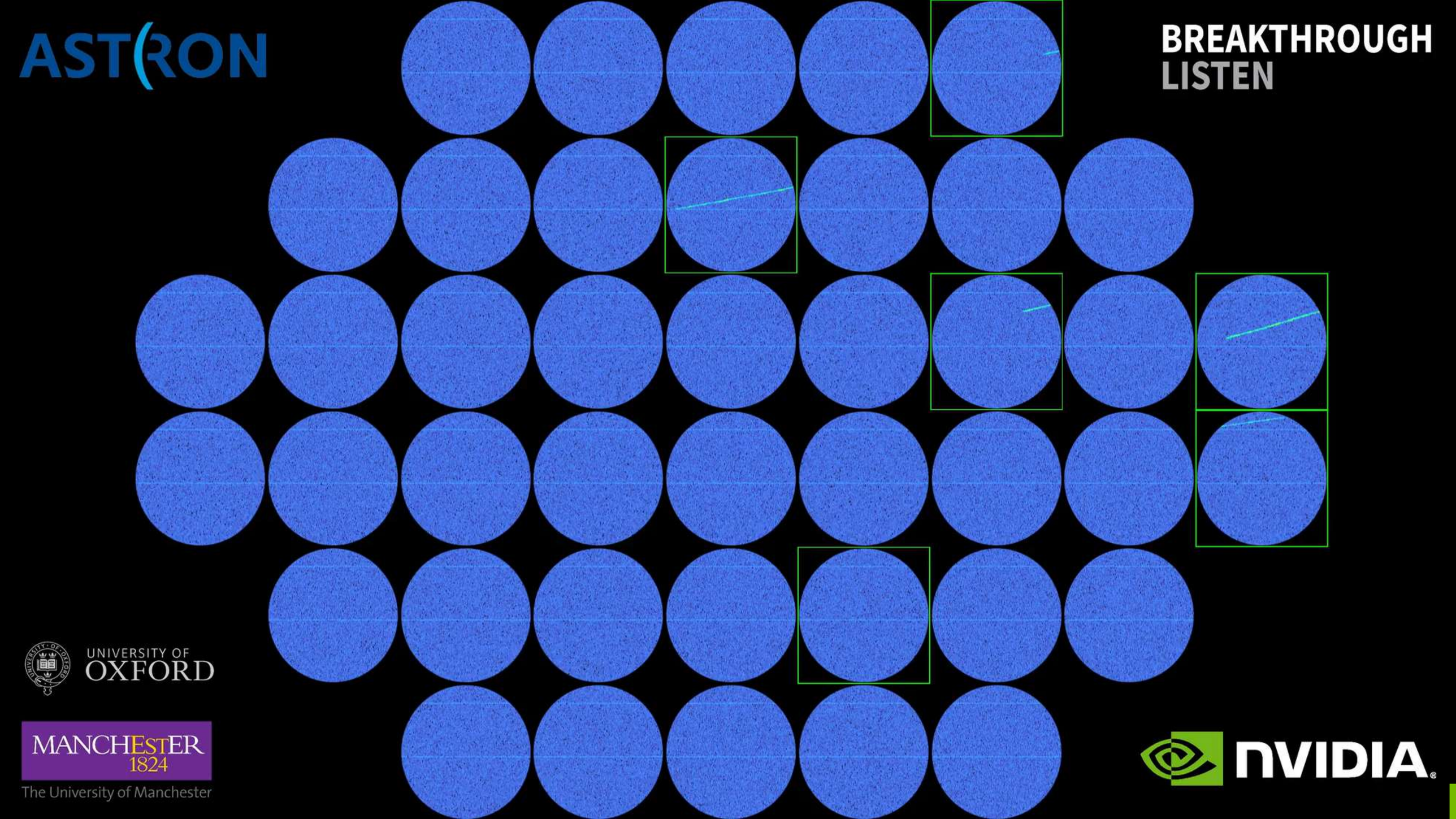
```
memory_regions:
- name: "Data_RX_CPU"
  kind: "huge"
  affinity: 0
  num_bufs: 51200
  buf_size: 64
- name: "Data_RX_GPU"
  kind: "device"
  affinity: 0
  num_bufs: 51200
  buf_size: 1000

interfaces:
- name: "rx_port"
  address: 00:05.1       # The BUS address of the interface doing Rx
  rx:
    flow_isolation: true
    queues:
    - name: "rq_q_0"
      id: 0
      cpu_core: 9
      batch_size: 10240
      memory_regions:
        - "Data_RX_GPU"
        - "Data_RX_GPU"
```

# Radio Astronomy

# Next Generation Digital Backend for Radio Astronomy

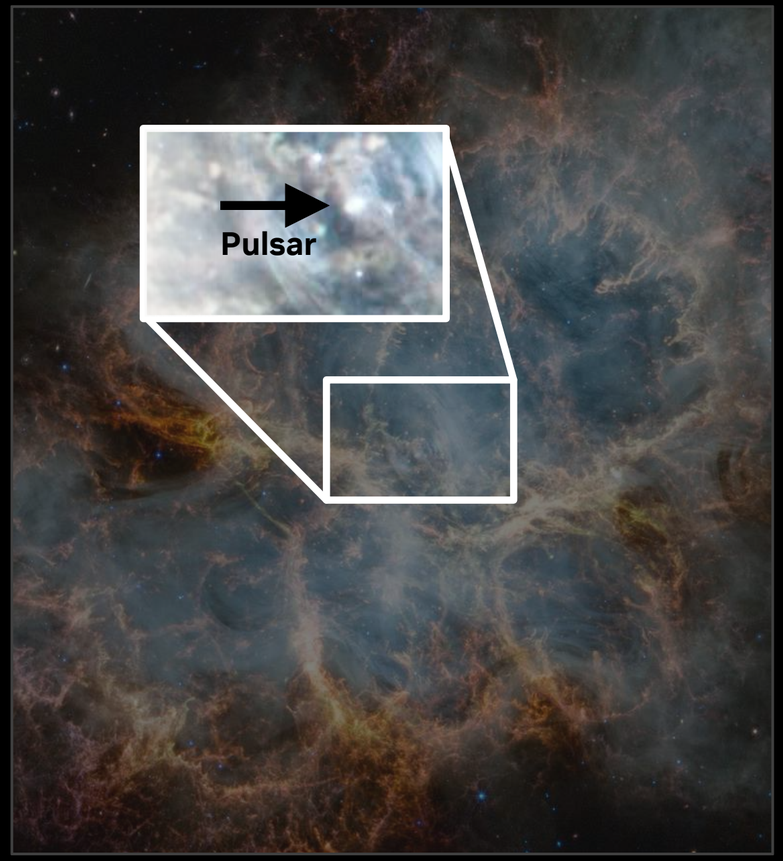
## Allen Telescope Array – Real Time AI Technosignature Search



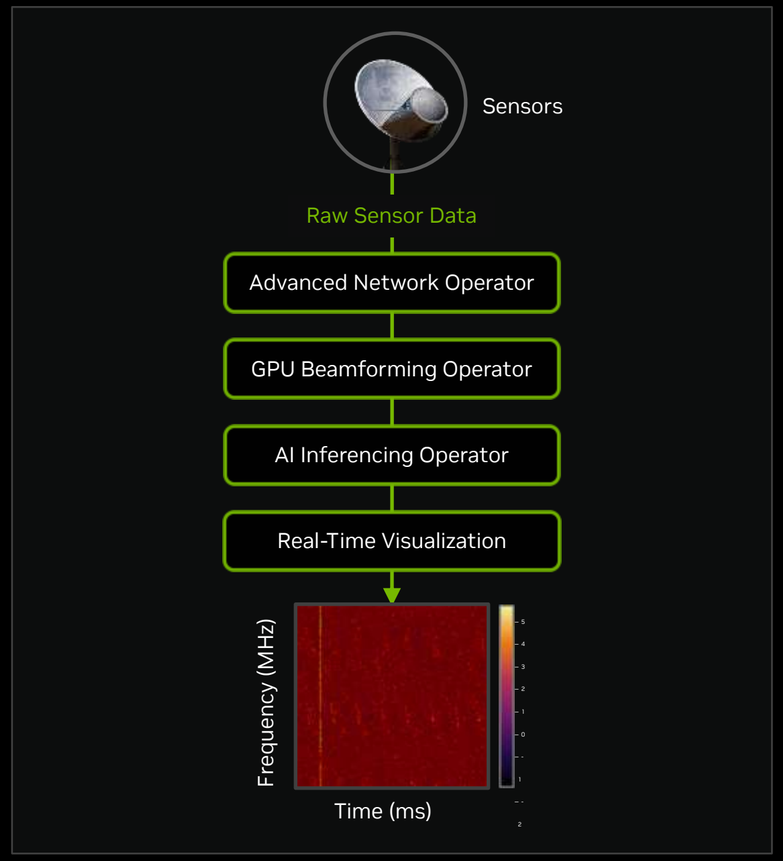

ASTRON
BREAKTHROUGH
LISTEN
UNIVERSITY OF
OXFORD
MANCHESTER
1824
The University of Manchester
NVIDIA

# First Real-Time Pure AI Detection of a Pulsar Using Raw Streaming Sensor Data

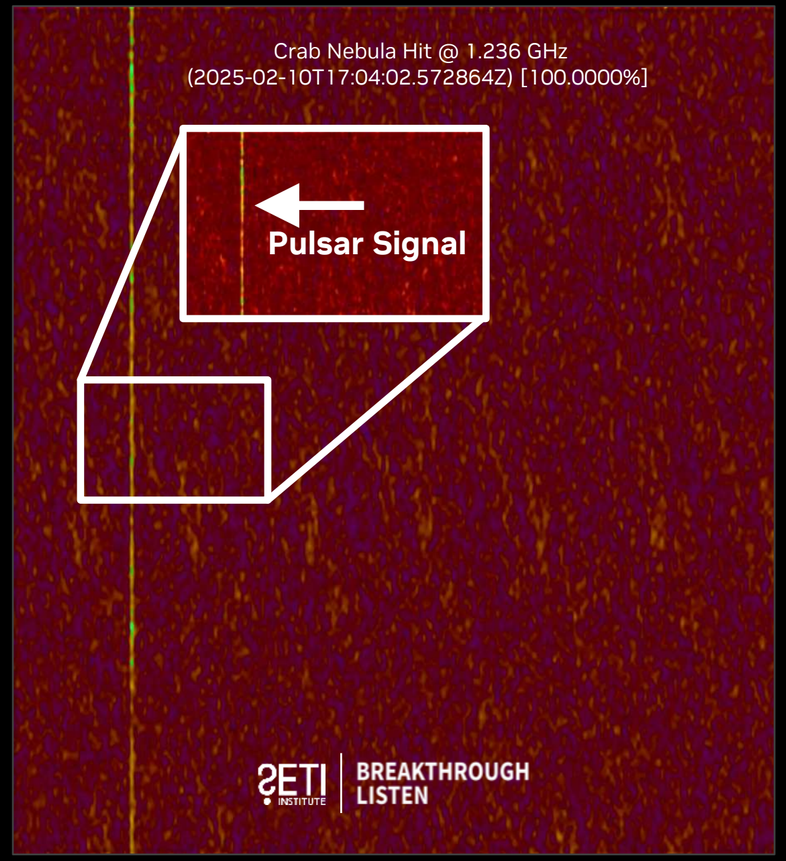
## Holoscan Enables Real-Time AI-Powered Sensor Workloads at the Edge




Pulsar in the Crab Nebula




Holoscan From Beamformer to AI Model

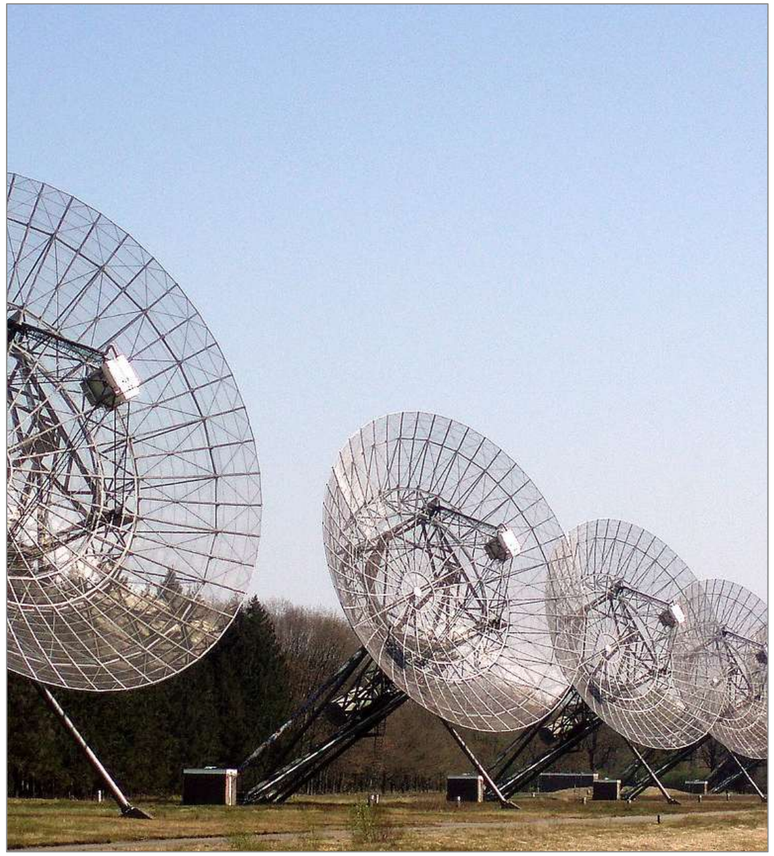



Signal Detection Beyond Noise

600x Speedup | 160x Faster than Real Time | 10x Reduction in False Alarms

# A New AI Digital Backend for Radio Astronomy

## GPU Digital Backend Adoption at Other Radio Observatories



**ASTRON – Westerbork Radio Observatory**

All-Sky Monitor for Billion+ Star Search

**NenuFar**

French SKA Pathfinder – Fast Radio Burst Detection

**GMRT**

India SKA Pathfinder – Fast Radio Burst and Pulsar Detection

# How We Did It
# … and How You Can Too!

# Technosignatures and ML Based Detection

## Higher Sensitivity, Lower False Positives, Faster Computational Speed

THE ASTROPHYSICAL JOURNAL

FREE ARTICLE

### Fast Radio Burst 121102 Pulse Detection and Periodicity: A Machine Learning Approach

Yunfan Gerry Zhang, Vishal Gajjar, Griffin Foster, Andrew Siemion, James Cordes, Casey Law, and Yu Wang

Published 2018 October 23 • © 2018. The American Astronomical Society. All rights reserved.

The Astrophysical Journal, Volume 866, Number 2

**Citation** Yunfan Gerry Zhang *et al* 2018 *ApJ* **866** 149

**DOI** 10.3847/1538-4357/aadf31

Article PDF | Article ePub

Authors | Figures | Tables | References

Article information

Abstract

We report the detection of 72 new pulses from the repeating fast radio burst FRB 121102 in Breakthrough Listen *C*-band (4–8 GHz) observations at the Green Bank Telescope. The new pulses were found with a convolutional neural network in data taken on 2017 August 26, where 21 bursts have been previously detected. Our technique combines neural network detection with dedispersion verification. For the current application, we demonstrate its advantage over a traditional brute-force dedispersion algorithm in terms of higher sensitivity, lower false-positive rates, and faster computational speed. Together with the 21 previously reported pulses, this observation marks the

**Zhang et Al**

72 New FRB Detections from Recorded Data at GBO

nature astronomy

Explore content | About the journal | Publish with us | Subscribe

nature › nature astronomy › articles › article

Article | Published: 30 January 2023

### A deep-learning search for technosignatures from 820 nearby stars

Peter Xiangyuan Ma, Cherry Ng, Leandro Rizk, Steve Croft, Andrew P. V. Siemion, Bryan Brzycki, Daniel Czech, Jamie Drew, Vishal Gajjar, John Hoang, Howard Isaacson, Matt Lebofsky, David H. E. MacMahon, Imke de Pater, Danny C. Price, Sofia Z. Sheikh & S. Pete Worden

*Nature Astronomy* **7**, 492–502 (2023) | Cite this article

**4785** Accesses | **26** Citations | **1755** Altmetric | Metrics

**Abstract**

The goal of the search for extraterrestrial intelligence (SETI) is to quantify the prevalence of technological life beyond Earth via their 'technosignatures'. One theorized technosignature is narrowband Doppler drifting radio signals. The principal challenge in conducting SETI in the radio domain is developing a generalized technique to reject human radiofrequency interference. Here we present a comprehensive deep-learning-based technosignature search on 820 stellar targets from the Hipparcos catalogue, totalling over 480 h of on-sky data taken with the Robert C. Byrd Green Bank Telescope as part of the Breakthrough Listen initiative.

**Ma et Al**

8 Previously Undetected SETI Signals of Interest


**Ma et Al**

Real Time End to End DL Algorithm for FRB Detection

*"I am making slides for the SciPy conference and have a talk on real time AI detection of fast radio bursts. I would like to create an image that says our paper is in review. This text should be prominently displayed but also feature images of construction cones and or radio astronomy things"*

NVIDIA

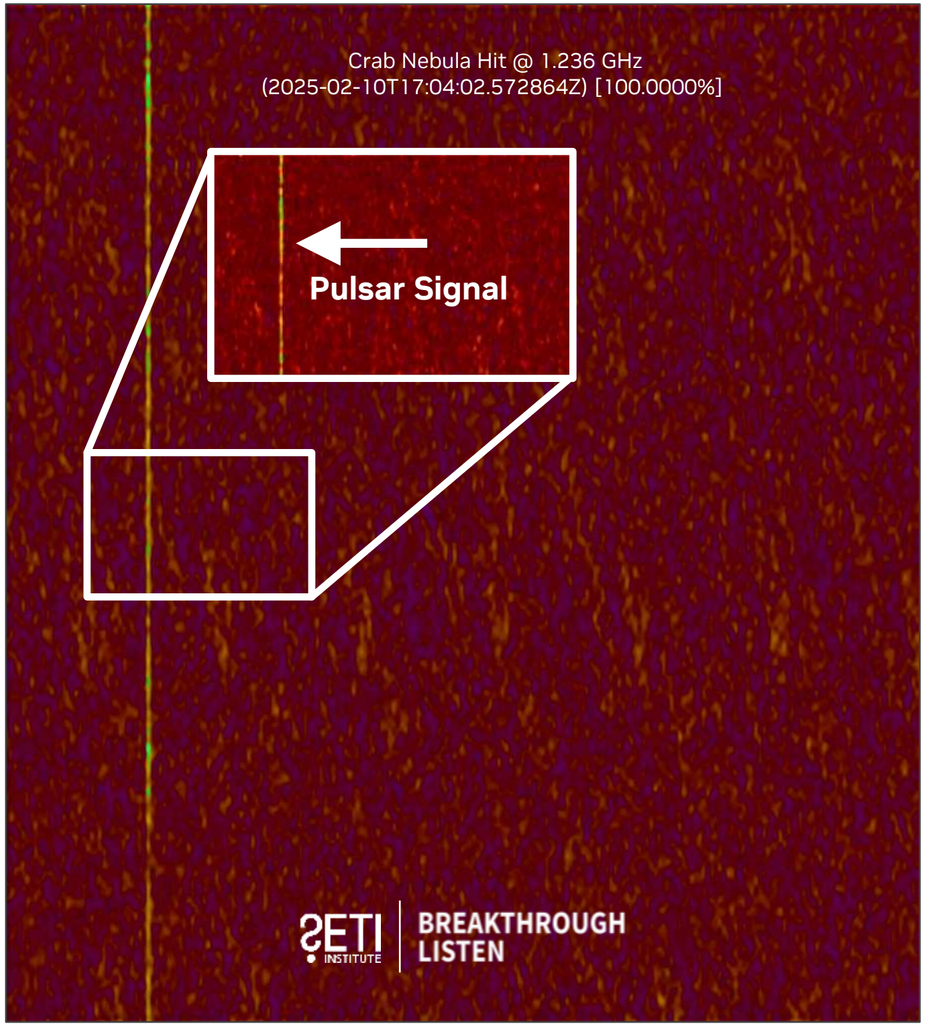
# Model Generation and Training

# Fast Radio Burst Neural Network (FRBNN)

## Real-Time End-to-End Deep Learning Algorithm for Fast Radio Burst Detection

*https://github.com/PetchMa/frbnn*

- Designed for Fast Radio Bursts (FRBs) detection.
- Lightweight model (82 MB) based on ResNet.
- Efficient TensorRT inference in real time.
- Simulation augmented training dataset:
  - Real observations from the Allen Telescope Array as base.
  - Simulated bursts synthetized via SciPy Signal and InjectFRB.
  - Resulted in 300 GB set containing 200K bursts.
- High recall rate throughout wide range of SNRs.
- Tested in a real-time setting at the Allen Telescope Array (ATA).
- Successfully identified the Crab Pulsar (PSR B0531+21) bursts.
- Paper (Ma et al. 2025) under review Astronomy & Astrophysics.



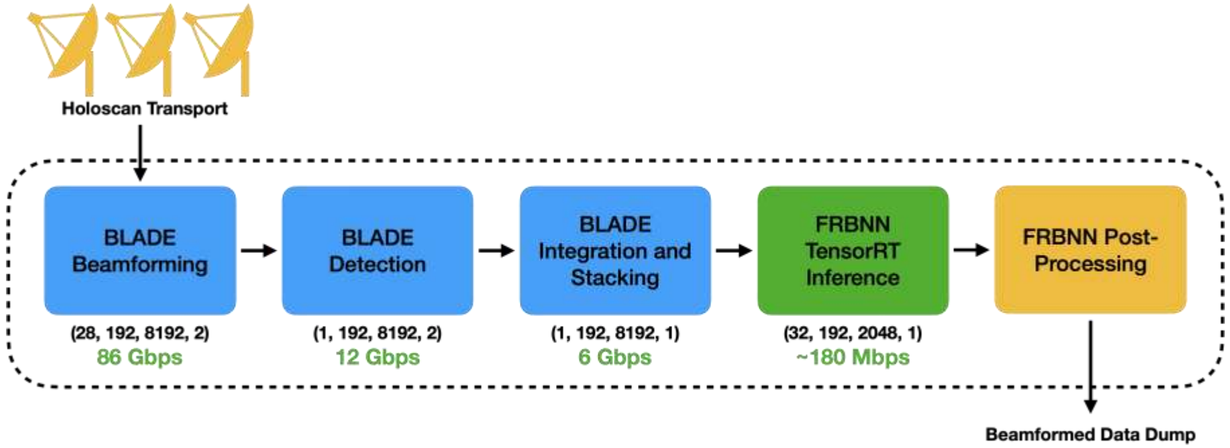
# Deployment

# Stelline

## Next Generation Digital Signal Processing Pipeline

*https://github.com/luigifcruz/stelline*

- Modular and composable real-time processing framework based on NVIDIA Holoscan.
- Aggregates custom Holoscan operators and glue code:
  - **BLADE**: Digital Signal Processing operators used for beamforming, correlation, etc.
  - **TensorRT**: Efficient inference in real-time on spectrogram data. Used by the FRBNN project.
  - **Transport**: Data reception via RDMA using the Advanced Network Operator.
  - **Filesystem**: Leverages NVIDIA GPUDirect Storage to store data to disk directly from the GPU memory. Support for HDF5 planned soon.
- Aims to replace the CPU-based pipeline at the Allen Telescope Array and other telescopes around the world.
- Customizable pipelines defined via YAML file.




(A, F, T, P) = A - Antennas, F - Channels, T - Time, P - Polarization

SETI INSTITUTE

# BLADE

## Breakthrough Listen Accelerated DSP Engine

*https://github.com/luigifcruz/blade*

- In-house Digital Signal Processing library.
- Focused on beamforming and correlation.
- Used at the Allen Telescope Array since 2022:
  - Process data from 28 antennas with more soon!
  - Each antenna produces ~3.0 GHz of bandwidth.
  - Equates to an aggregated 1.4 Tbps of data.
- Acts as a common interface between astronomy libraries.
- Accelerated via CUDA kernels compiled just-in-time.
- Provides Python bindings for easy prototyping.

```python
# Import the blade library
import blade as bl

# Define the synchronous pipeline class
@bl.runner
class Pipeline:
    # Initialize the pipeline with shape and configuration
    def __init__(self, shape, config):
        # Create input and output buffers with the specified shape
        self.input.buf = bl.array_tensor(shape, dtype=bl.cf32)
        self.output.buf = bl.array_tensor(shape, dtype=bl.cf32)

        # Initialize the polarizer module with the given configuration and input buffer
        self.module.polarizer = bl.module(bl.polarizer, config, self.input.buf)

    # Transfer data from the provided buffer to the input buffer
    def transfer_in(self, buf):
        self.copy(self.input.buf, buf)

    # Transfer data from the polarizer's output to the output buffer and then to the provided buffer
    def transfer_out(self, buf):
        self.copy(self.output.buf, self.module.polarizer.get_output())
        self.copy(buf, self.output.buf)

# Define the shape and configuration for the pipeline
shape = (2, 192, 8750, 2)
config = {
    'inputPolarization': bl.pol.xy,
    'outputPolarization': bl.pol.lr,
}

# Create an instance of the pipeline
pipeline = Pipeline(shape, config)

# Create host input and output buffers with the specified shape and device
host_input = bl.array_tensor(shape, dtype=bl.cf32, device=bl.cpu)
host_output = bl.array_tensor(shape, dtype=bl.cf32, device=bl.cpu)

# Execute the pipeline synchronously with the host input and output buffers
pipeline(host_input, host_output)
```

SETI INSTITUTE

# Hardware Architectures for HPC and AI at the Edge

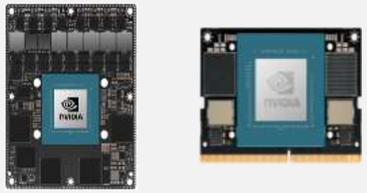
# Hardware Architectures for HPC and AI at the Edge

## Different Options for Different Sensor Requirements

### Embedded Edge

**Jetson Thor**
40-130W
128GB
2070 TFTOPS (FP4)

**Jetson Orin**
7-65W
Up to 64GB
34 – 275 TOPS (INT8)

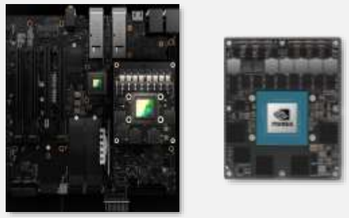


**< 100 GbE I/O Thor**
**< 10 GbE I/O Orin**

### Industrial Edge

**IGX Thor**
130W – 430W
128GB + 96GB
2070 – 5581 TFLOPS (FP4)

**IGX Orin**
15-125W
64GB + 48GB
248 - 1705 TOPS (INT8)



**< 400 GbE I/O Thor**
**< 200 GbE I/O Orin**

### Enterprise Edge

**Edge GPU**
**(RTX 6000 Pro/ L40S/L4)**
40W-600W
Up to 96 GB
Up to 3700 TFLOPS (FP4)

**DGX Spark/ Station**
TBD
128 GB / 784 GB
1 PFLOPS (FP4) / 20 PFLOPS (FP4)

**< 400 GbE**
**Scalable per NIC/GPU**

### Data Center / HPC Edge

**GH 200**
450W – 1000W
624GB
4 PF AI Perf

**~1 Tbps I/O**

# Getting Started

# Getting Started with Holoscan

## Holoscan References

https://github.com/nvidia-holoscan/holoscan-sdk

```
docker pull nvcr.io/nvidia/clara-holoscan/holoscan:v3.6.0-dgpu
```

```
pip install holoscan
conda install -c conda-forge holoscan
```

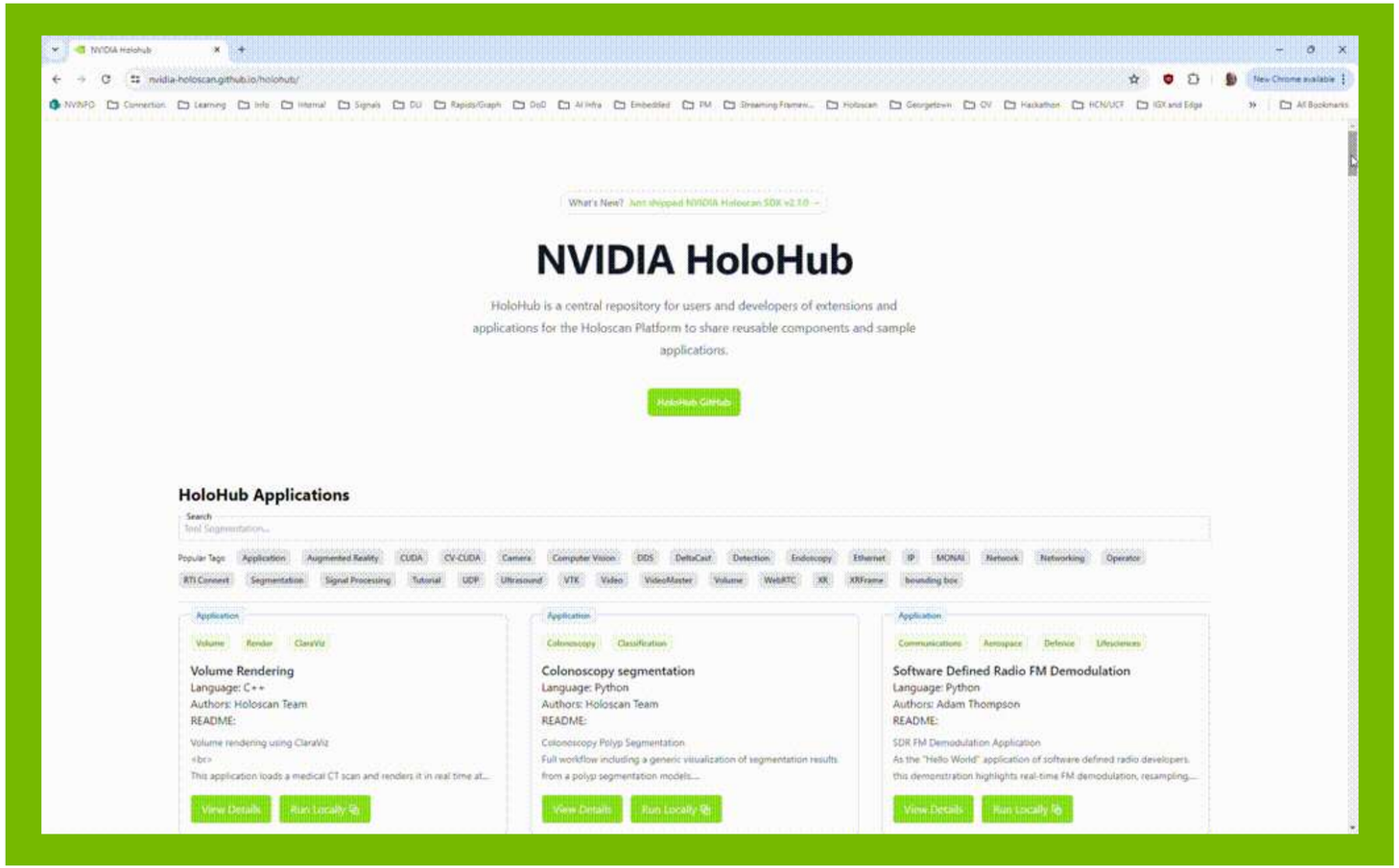
Debian Packages available on NGC

https://docs.nvidia.com/clara-holoscan/sdk-user-guide/index.html

# Holoscan Reference Applications

Sample Holoscan Applications, Tutorials, and Helpful Operators

GitHub




**IO:** UDP Ethernet, Lidar, High Speed Cameras, SDR, SAR, 3D Volumes
**AI:** Segmentation, Tracking, AR/VR, LLMs
**Tools/Tutorial:** MATLAB, MONAI, Playground on AWS, Scaling Apps