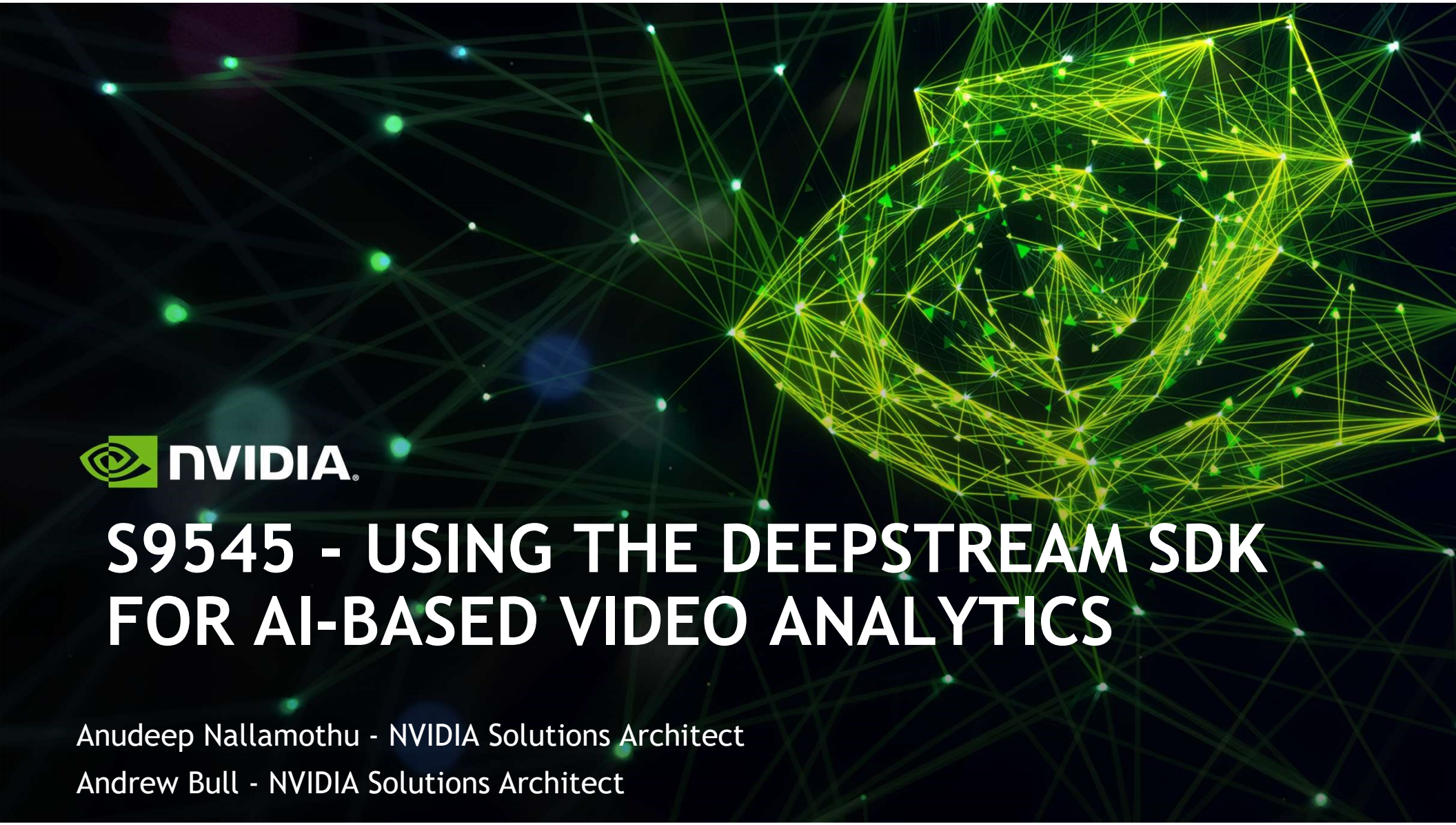


S9545 - USING THE DEEPSTREAM SDK FOR AI-BASED VIDEO ANALYTICS

Anudeep Nallamothu - NVIDIA Solutions Architect

Andrew Bull - NVIDIA Solutions Architect

- Realtime Streaming Video Analytics
- Framework for Analyzing Video
- Understand the Basics: DeepStream SDK 3.0
- Hardware Platforms
- An Overview of TensorRT 5.0
- Transfer Learning Toolkit
- Build with DeepStream: Example Applications
- Getting Started Resources

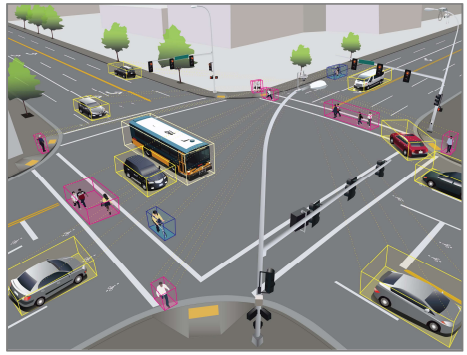

REALTIME STREAMING
VIDEO ANALYTICS

REALTIME STREAMING VIDEO ANALYTICS FROM EDGE TO CLOUD

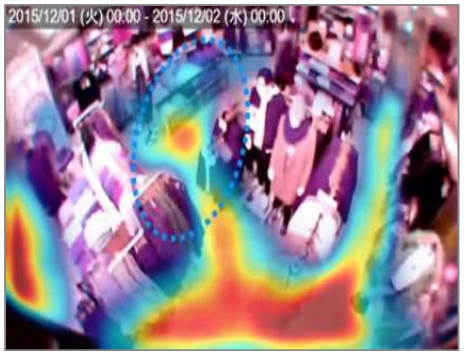
Access Control

Managing operations

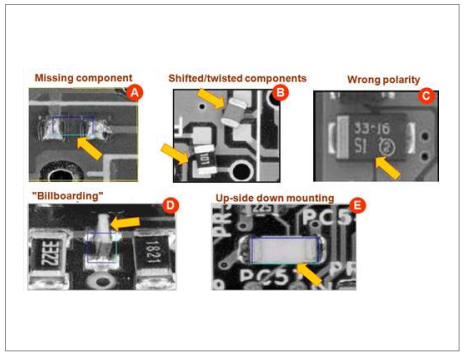
Parking Management



Traffic Engineering



Retail Analytics



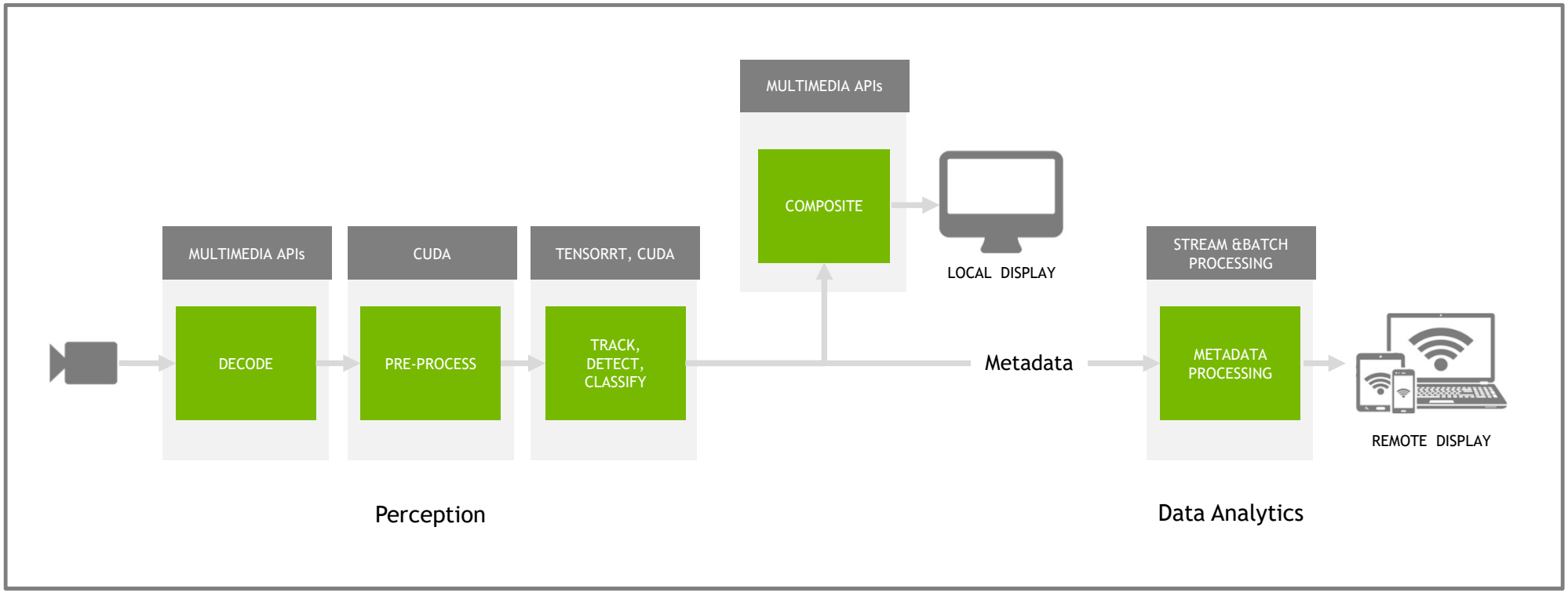
Optical Inspection

Managing Logistics

Content Filtering


FRAMEWORK FOR
ANALYZING VIDEO

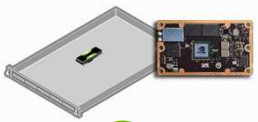
FRAMEWORK FOR ANALYZING VIDEO



DEEPSTREAM FOR AI APPLICATION PERFORMANCE AND SCALE

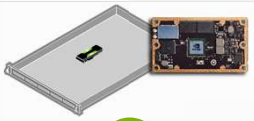
Scalability

Perception
- Platform specific APIs
- Streams: Multi (Tesla), single(Jetson)



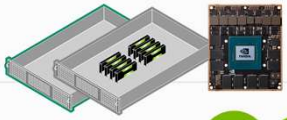
v1.0

Perception - edge to cloud
- Unified APIs across platforms
- Multi-streams/ multi-DNNs
- Custom graphs

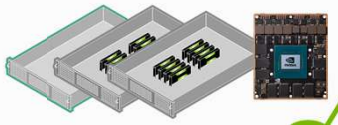


v2.0

Perception and Analytics
- Multi-GPU containerized applications
- 360D cameras
- Dynamic stream management
- IOT services



v3.0



NEXT

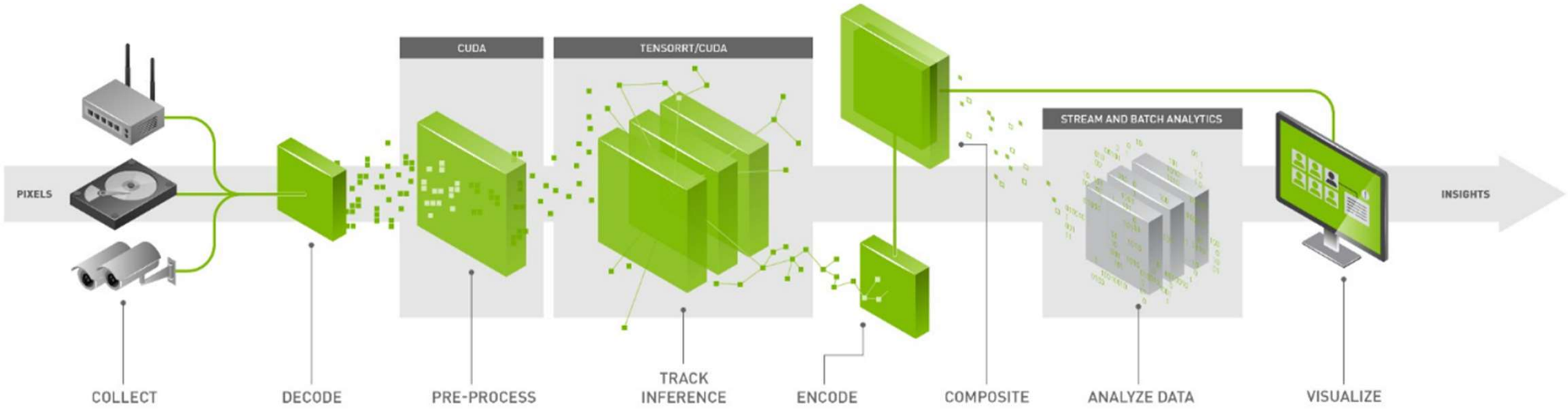
Solution framework
- Optical flow
- Remote display
- Multi-GPU dynamic orchestration
- Indexed video storage and retrieval
- Workflow templates for full solutions

DeepStream Next - POR can change


DEEPSTREAM 3.0

DEEP LEARNING FOR IVA

End-to-end workflow

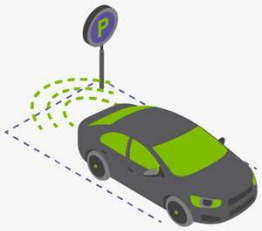


DEEPSTREAM SDK

USER APPLICATIONS

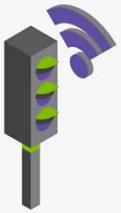
ACCESS CONTROL

SMART PARKING



RETAIL ANALYTICS/CHECKOUT

INTELLIGENT TRAFFIC SYSTEMS



LAW ENFORCEMENT

DEEPSTREAM SDK

PLUG-INS (BUILD WITH OPEN SOURCE, THIRD PARTY, NVIDIA)

- DNN Inference/TensorRT Plug-Ins
- Communications Plug-Ins
- Video/Image Capture and Processing Plug-Ins
- Third-Party Library Plug-Ins

ANALYTICS: MULTI-CAMERA, MULTI-SENSOR FRAMEWORK

- DeepStream in Containers, Multi-GPU Orchestration
- Tracking & Analytics Across Large Scale Multi-Camera
- Streaming and Batch Analytics
- Event Fabric

DEVELOPMENT TOOLS

- End-to-End Reference Applications
- App Building/Configuration Tools
- Plug-In Templates and Adaptation Guides
- Profiling and Performance Tuning

TENSORRT | MULTIMEDIA APIs / VIDEO CODEC SDK | IMAGING & DEWARPING LIBRARY | METADATA & MESSAGING | NVIDIA CONTAINERS | MESSAGE BUS CLIENTS | MULTI-CAMERA TRACKING LIB

LINUX, CUDA

PERCEPTION INFRA - JETSON, TESLA SERVER (EDGE AND CLOUD) | ANALYTICS INFRA - EDGE SERVER, NGC, AWS, AZURE

NVIDIA IVA PLATFORM

Deploy from the edge to the cloud

EDGE / ON-PREMISE
Inference

CORE/CLOUD
Training and Inference

Camera

NVR / APPLIANCE

SERVER

Data center

DEEPSTREAM • TENSORRT • JETPACK

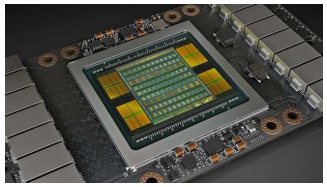
JETSON

QUADRO / TESLA

TESLA / DGX

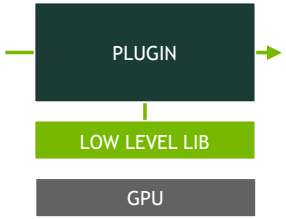
WHAT'S NEW IN DEEPSTREAM 3.0

LATEST GPUs - TESLA T4, JETSON XAVIER



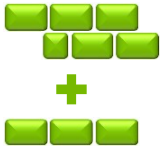
TensorRT 5, CUDA 10

NEW PLUGINS



Increased capability and throughput

DYNAMIC STREAM MANAGMENT



Add, remove, modify streams on the fly

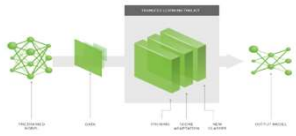
CONNECT EDGE TO CLOUD

Stream and Batch Analytics on Metadata

EASY TO SCALE AND MANAGE

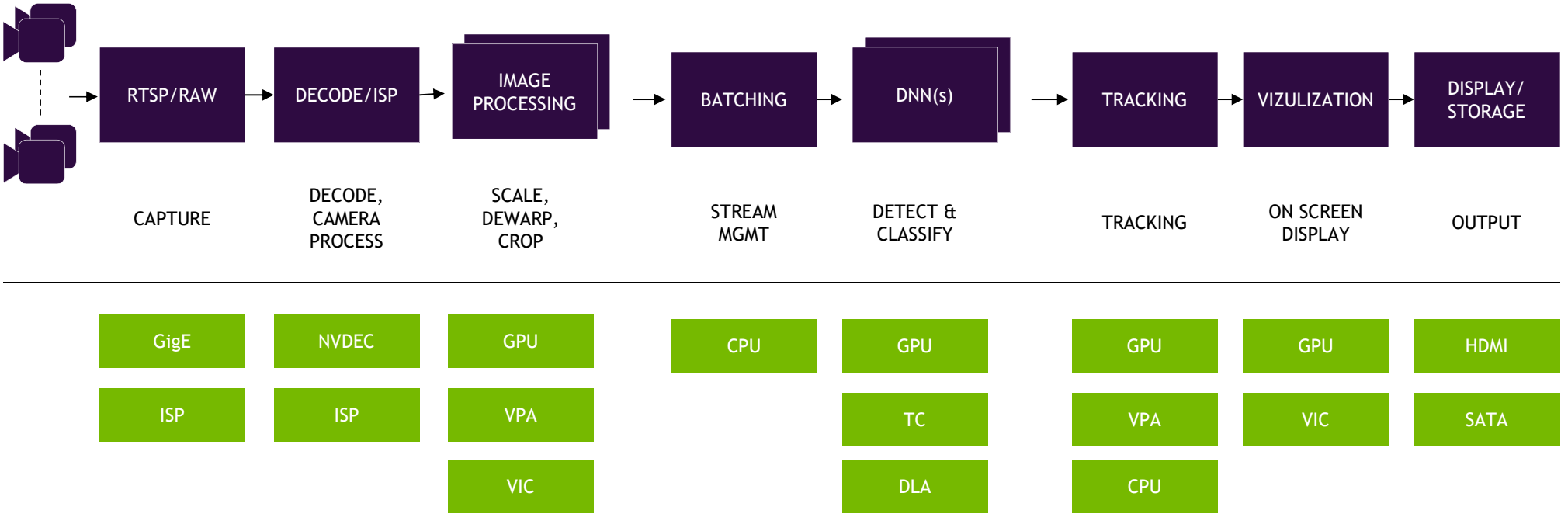
Deploy in Docker Containers

HIGH EFFIENCY AND THROUGHPUT WITH TLT



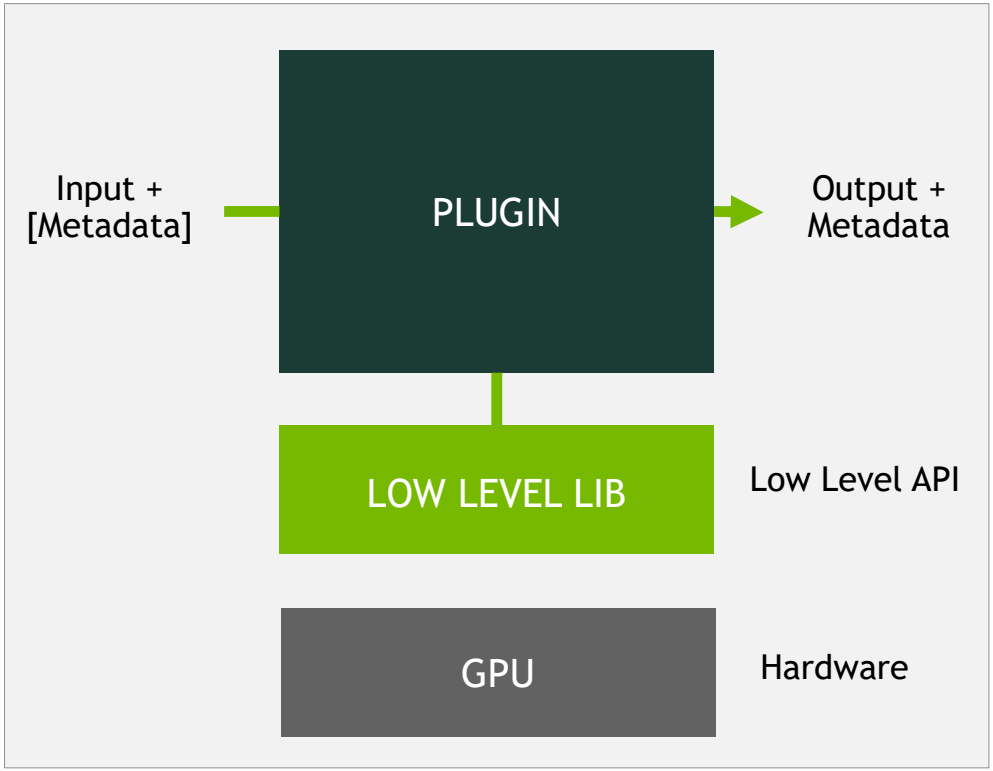
TLT model files are plug-n-play

DEEPSTREAM STREAMING ARCHITECTURE



DEEPSTREAM BUILDING BLOCK

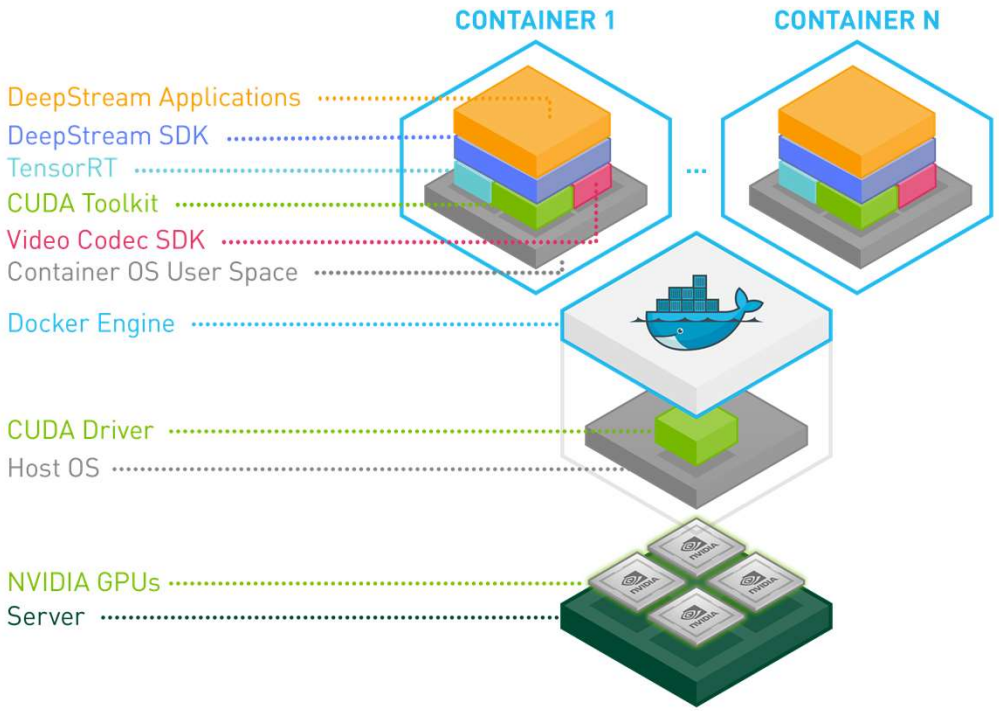
- A plugin model based pipeline architecture
- Graph-based pipeline interface to allow high-level component interconnect
- Heterogenous processing on GPU and CPU
- Hides parallelization and synchronization under the hood
- Inherently multi-threaded



NVIDIA-ACCELERATED PLUGINS

Plugin Name	Functionality
gst-nvvideocodecs	Accelerated video decoders
gst-nvstreammux	Stream aggregator - muxer and batching
gst-nvinfer	TensorRT based inference for detection & classification
gst-nvtracker	Reference KLT tracker implementation
gst-nvosd	On-Screen Display API to draw boxes and text overlay
gst-tiler	Renders frames from multi-source into 2D grid array
gst-eglglessink	Accelerated X11 / EGL based renderer plugin
gst-nvvidconv	Scaling, format conversion, rotation
Gst-nvdewarp	Dewarping for 360 Degree camera input
Gst-nvmsgconv	Meta data generation
Gst-nvmsgbroker	Messaging to Cloud

SCALE WITH DEEPSTREAM IN DOCKER



NVIDIA GPU CLOUD

Discover GPU-Accelerated Containers

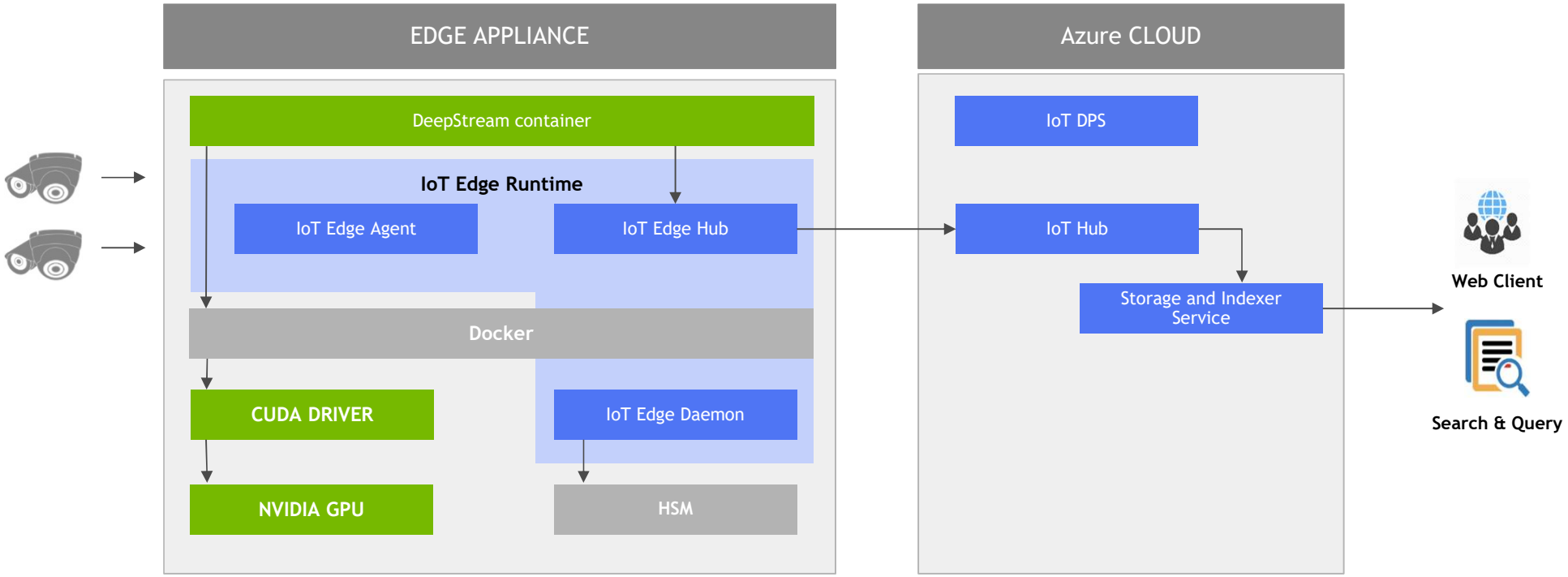
Innovate in Minutes, Not Weeks

Stay Up to Date

https://www.nvidia.com/en-us/gpu-cloud/


DEEPSTREAM IOT

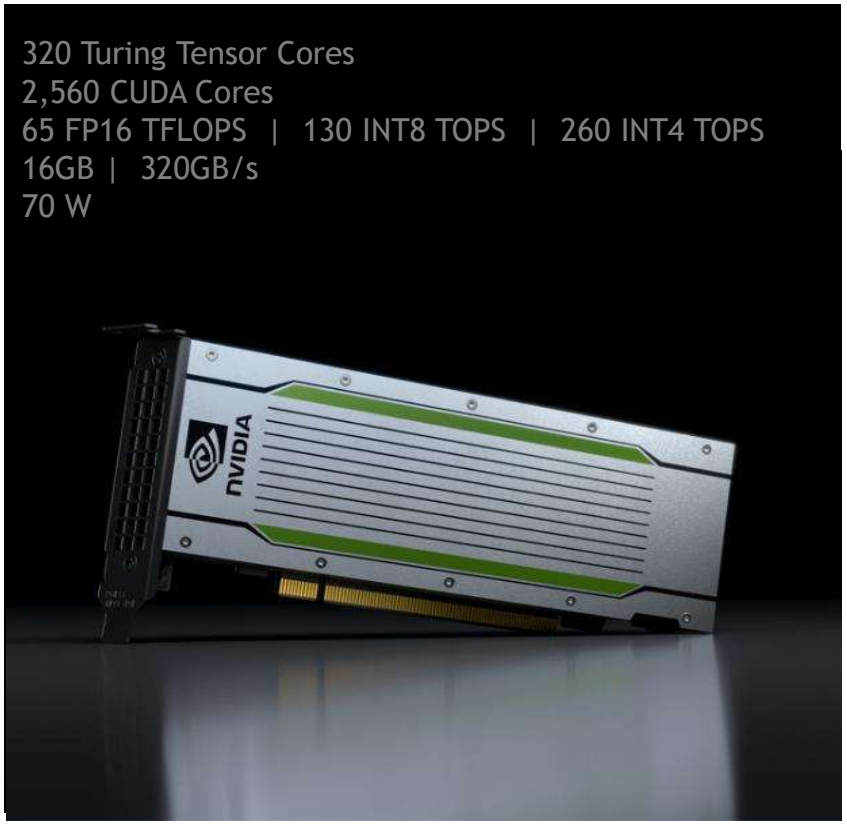
DEEPSTREAM WITH AZURE IOT

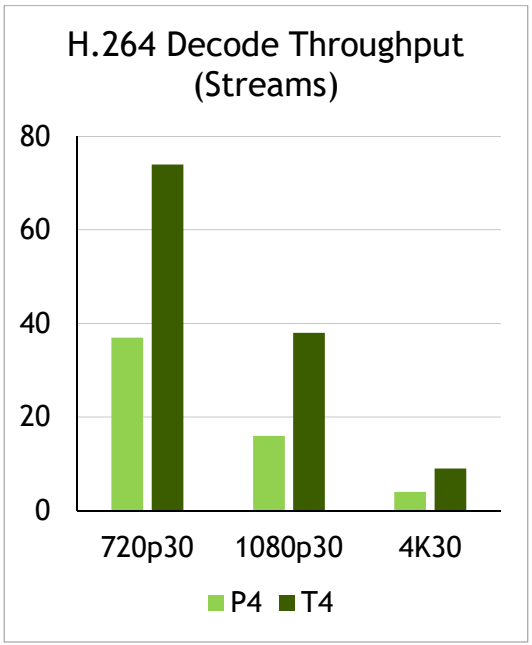


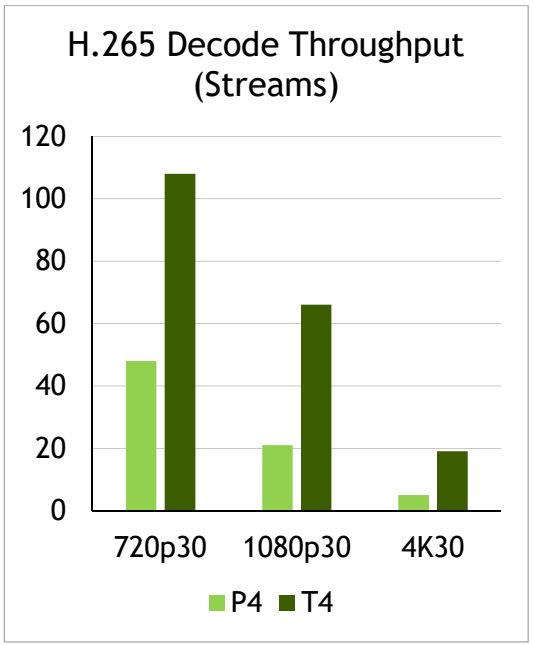

HARDWARE
PLATFORMS

NVIDIA T4 UNIVERSAL INFERENCE ACCELERATOR

320 Turing Tensor Cores
2,560 CUDA Cores
65 FP16 TFLOPS | 130 INT8 TOPS | 260 INT4 TOPS
16GB | 320GB/s
70 W







THE JETSON FAMILY

From AI at the Edge to Autonomous Machines

JETSON NANO
5 - 10W
0.5 TFLOPS (FP16)
45mm x 70mm
$129 AVAIABLE IN Q2

JETSON TX2
7 – 15W
1.3 TFLOPS (FP16)
50mm x 87mm
$299 - $749

JETSON AGX XAVIER
10 – 30W
10 TFLOPS (FP16) | 32 TOPS (INT8)
100mm x 87mm
$1099

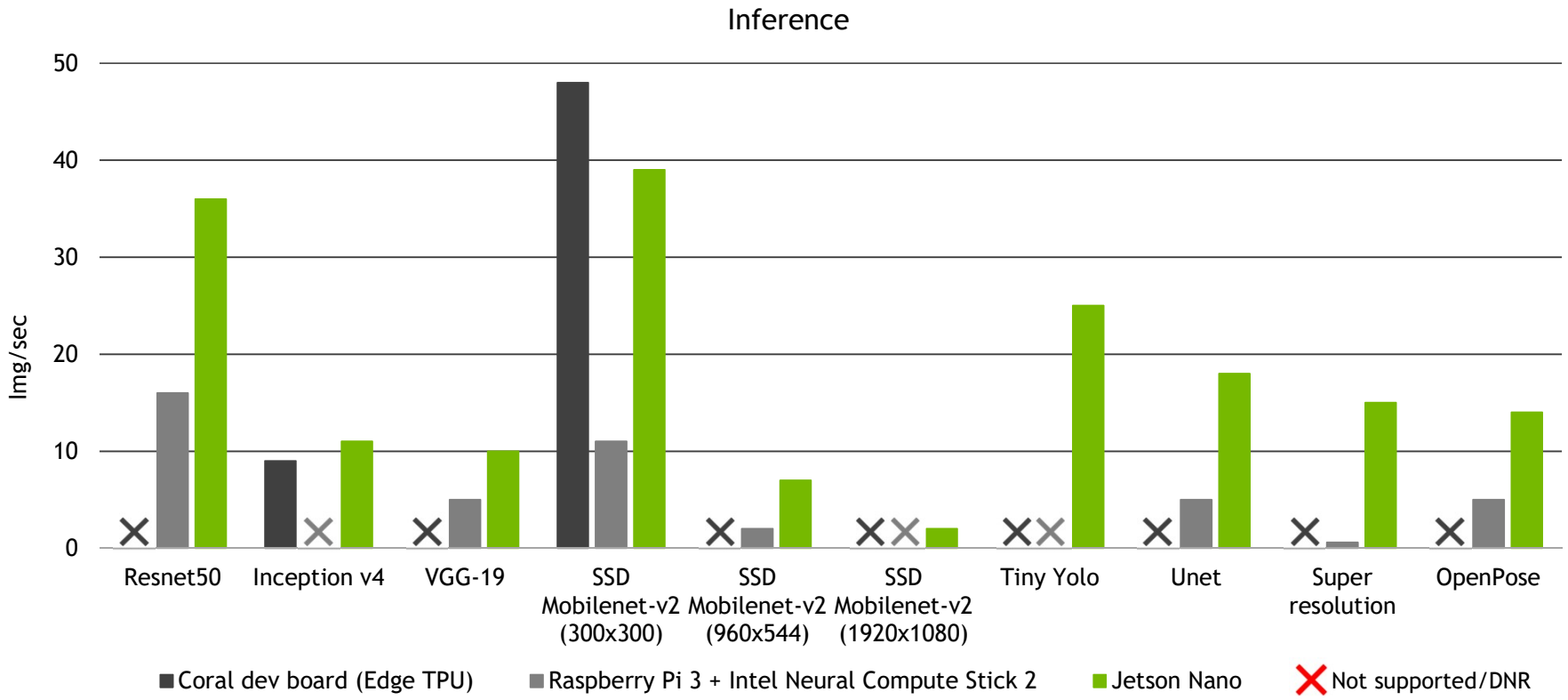
AI at the edge

Fully autonomous machines

Multiple devices - Same software

	JETSON NANO	JETSON TX2		JETSON AGX XAVIER
GPU	128 Core Maxwell 0.5 TFLOPs (FP16)	256 Core Pascal 1.3 TFLOPS (FP16)		NVIDIA Volta architecture with 512 NVIDIA CUDA cores and 64 Tensor cores
CPU	4 core ARM A57 @ 1.43 GHz	6 core Denver and A57 @ 2GHz		8-core ARM v8.2 64-bit CPU, 8 MB L2 + 4 MB L3
Memory	4 GB 64 bit LPDDR4 25.6 GB/s	4 GB 128 bit LPDDR4 51 GB/s	8 GB 128 bit LPDDR4 58 GB/s	16 GB 256-bit LPDDR4x
Storage	16 GB eMMC	16 GB eMMC	32 GB eMMC	32 GB eMMC 5.1
Video Encode	4K @ 30 | 4x 1080p @ 30 | 8x 720p @ 30 (H.264/H.265)	2x 4K @ 60 | 4x 4K @ 30| 14x 1080p @ 30 (H.264/H.265)		2x1000MP/sec | 4x 4K @ 60 (HEVC) 8x 4K @ 30 (HEVC)| 16x 1080p @ 60 (HEVC) 32x 1080p @ 30 (HEVC)
Power mode	5W|10W	7.5W|15W	7.5W|15W	10W|20W
Video Decode	4K @ 60 | 2x 4K @ 30 | 8x 1080p @ 30 | 16x 720p @ 30 | (H.264/H.265)	2x 4K @ 60| 4x 4K @ 30| 14x 1080p @ 30 (H.264/H.265)		2x1500MP/sec| 2x 8K @ 30 (HEVC) 6x 4K @ 60 (HEVC) | 12x 4K @ 30 (HEVC) 26x 1080p @ 60 (HEVC) |52x 1080p @ 30 (HEVC)
Camera	12 (3x4 or 4x2) MIPI CSI-2 DPHY 1.1 lanes (1.5 Gbps)	12 (3x4 or 6x2) MIPI CSI-2 D-PHY 1.2 lanes (30 Gbps)		16 lanes MIPI CSI-2 | 8 SLVS-EC D-PHY 1.2 (2.5Gb/s per pair, total up to 40 Gbps) C-PHY 1.1(2.5Gsym/s per trio, total up to 109 Gbps)
WiFi/BT	Requires external chip	Requires external chip	Onboard	Requires external chip
Display	HDMI 2.0 or DP1.2 | eDP 1.4 | DSI (1 x2) 2 simultaneous	HDMI 2.0 or DP 1.2 | eDP 1.4 | DSI (2 x4) 3 simultaneous		Three multi-mode DP 1.2a/eDP 1.4/HDMI 2.0 a/b No DSI support
UPHY	1 x1/2/4 PCIE | 1 USB 3.0	1+ 1 x4 or 1+1+1 x1/x2 PCIe or 3xUSB 3.0		16 lanes PCIe Gen 4 1x8 + 1x4 + 1x2 + 2x1
SATA	None	1x		SATA through PCIe x1 Bridge
Power mode	5W|10W	7.5W|15W	7.5W|15W	10W|20W
USB OTG	Not supported	Not Supported	Not Supported	Not Supported
Mechanical	69.6mm x 45mm 260 pin edge connector, No TTP	87mm x 50mm 400 pin connector, Integrated TTP		100 mm x87 mm 699-pin connector

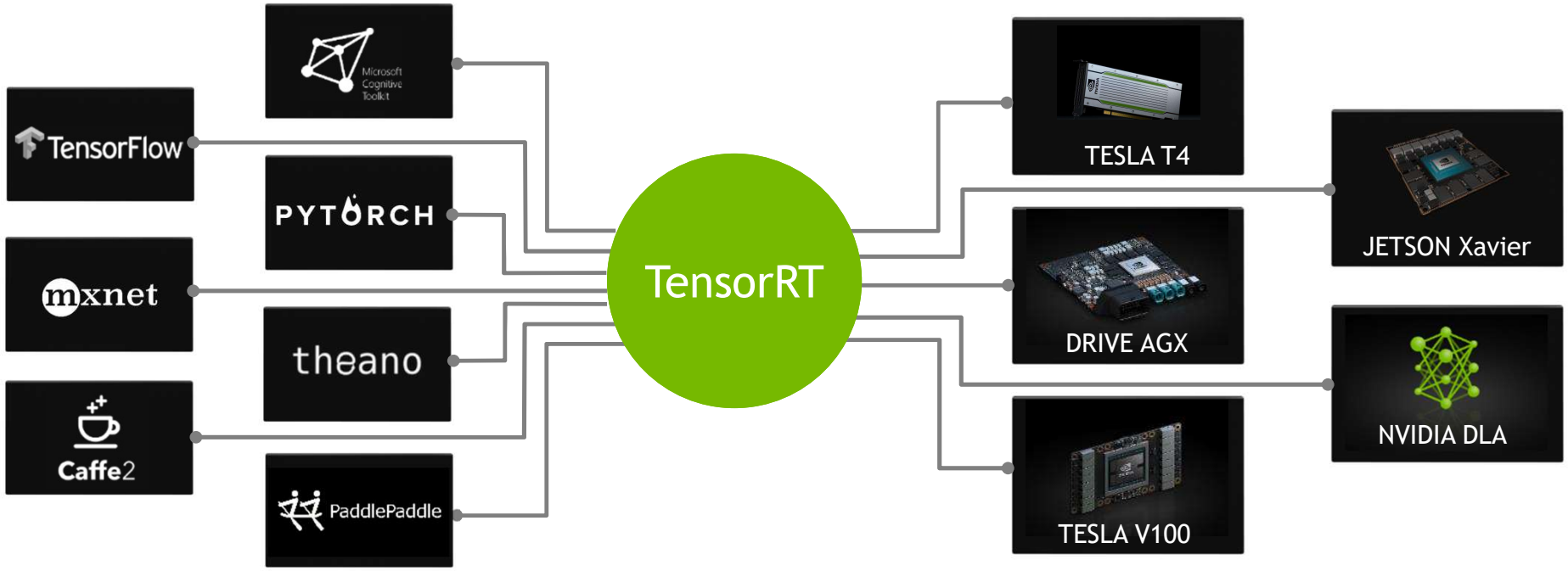
JETSON NANO RUNS MODERN AI




TENSORRT

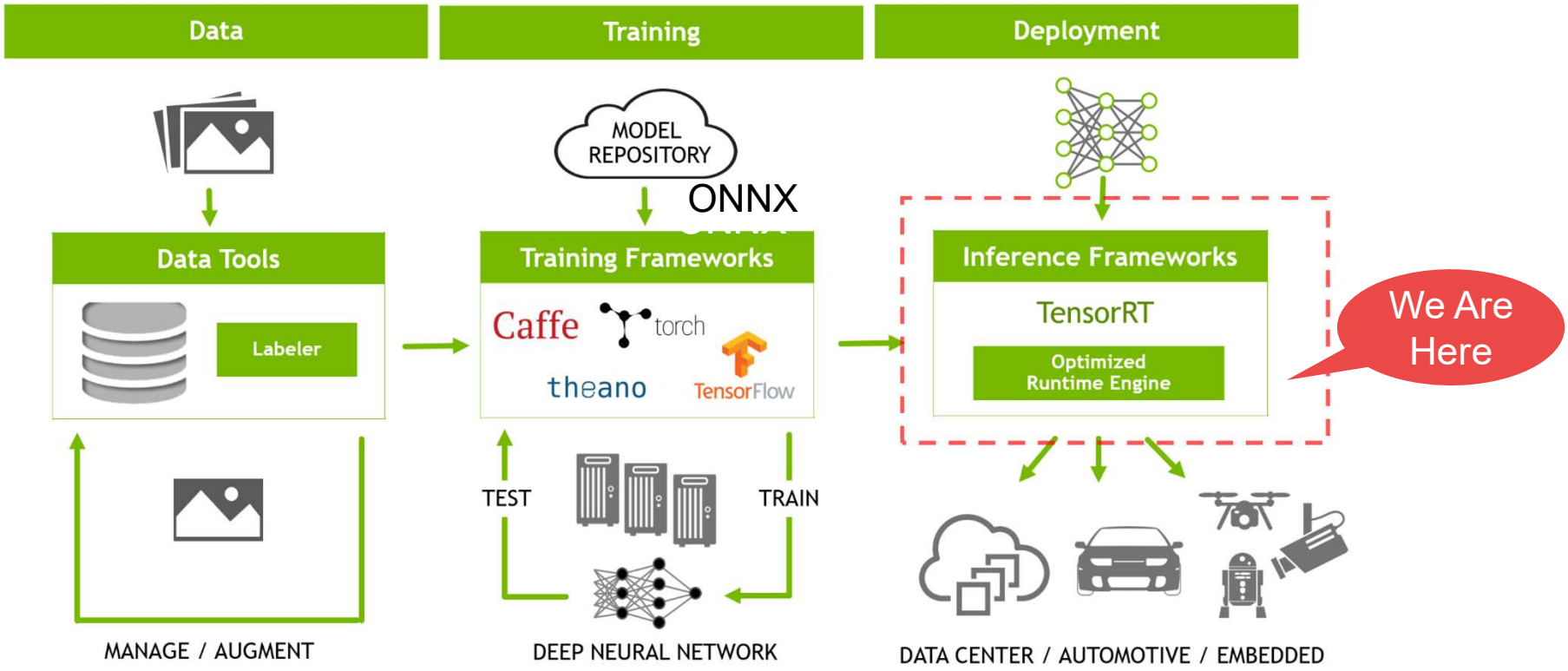
NVIDIA TensorRT

From Every Framework, Optimized For Each Target Platform



TENSORRT OVERVIEW

High-performance Deep Learning Inference Engine for Production Deployment

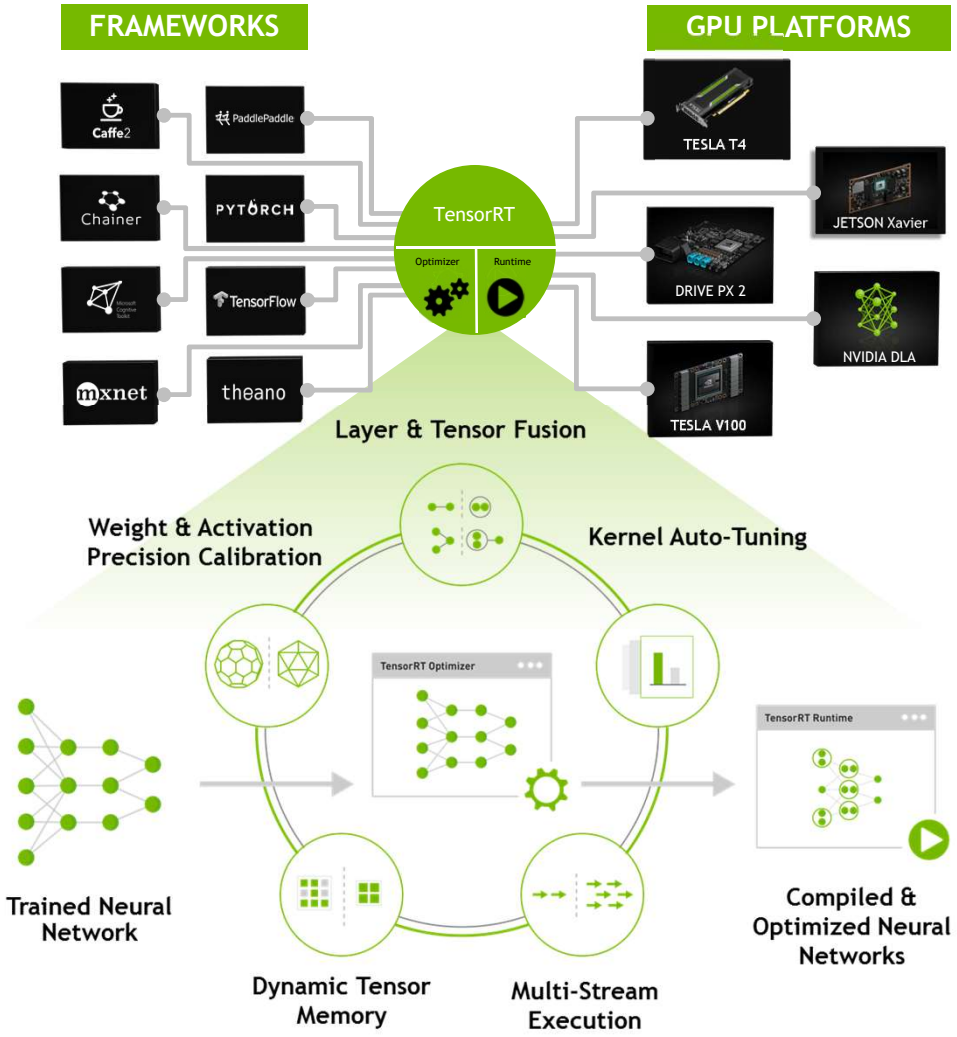


Inference Optimizer and Runtime

NVIDIA TensorRT 5

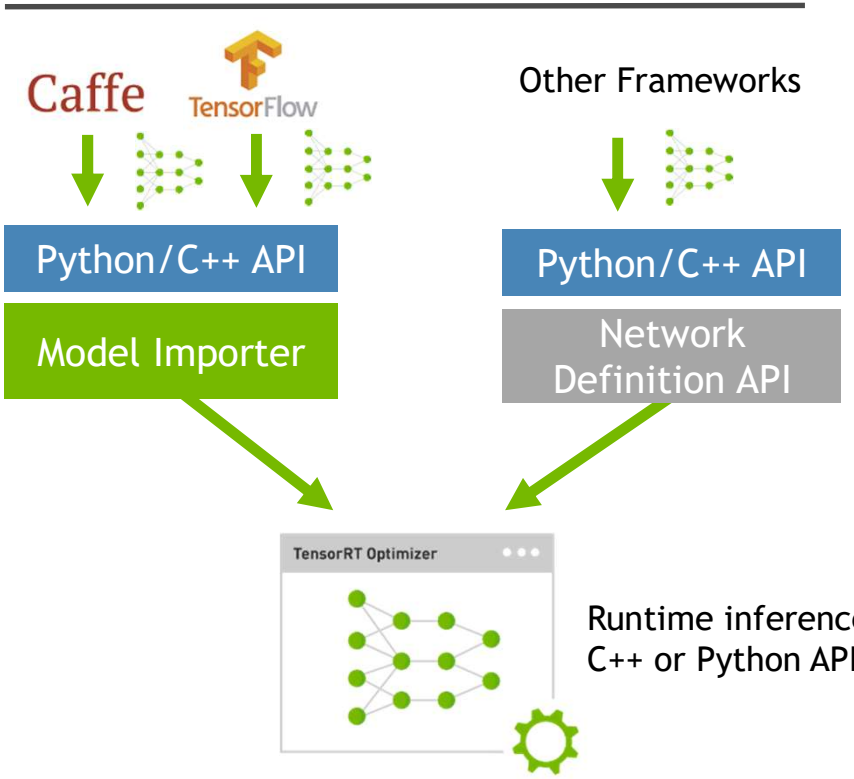
- Data center, embedded & automotive
- In-framework support for TensorFlow
- Support for all other frameworks and ONNX
- TensorRT inference server microservice with Docker and Kubernetes integration
- New layers and APIs
- New OS support for Windows and CentOS

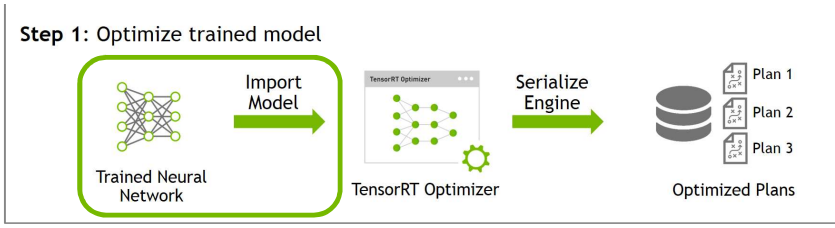
*New in TRT5



MODEL IMPORTING

- AI Researchers
- Data Scientists



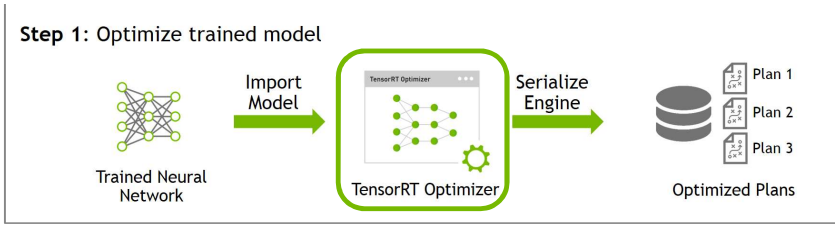


Example: Importing a TensorFlow model

```
import tensorrt as trt
import uff
from tensorrt.parsers import uffparser

G_LOGGER = trt.infer.ConsoleLogger(trt.infer.LogSeverity.INFO)

uff_model = uff.from_tensorflow_frozen_model("frozen_model.pb",
                                             "dense_2/Softmax")

parser = uffparser.create_uff_parser()
parser.register_input("input_1", (3,224,224),0)
parser.register_output("dense_2/Softmax")

engine = trt.utils.uff_to_trt_engine(G_LOGGER,
                                     uff_model,
                                     parser,
                                     INFERENCE_BATCH_SIZE,
                                     1<<20,
                                     trt.infer.DataType.FLOAT)

runtime = trt.infer.create_infer_runtime(G_LOGGER)
context = engine.create_execution_context()
```

developer.nvidia.com/tensorrt

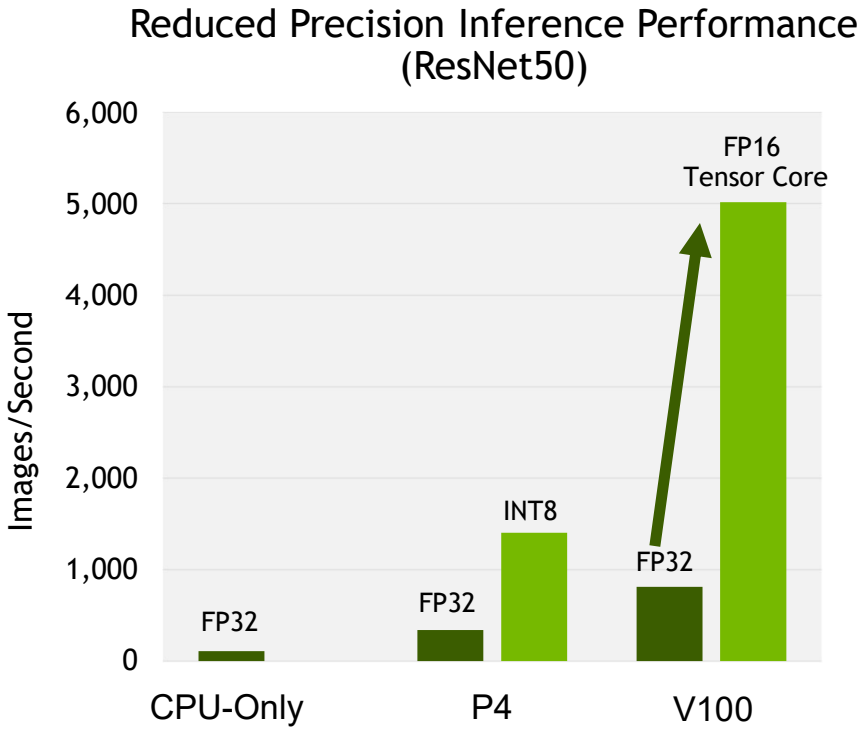
FP16, INT8 PRECISION CALIBRATION



	FP32 Top 1	INT8 Top 1	Difference
Googlenet	68.87%	68.49%	0.38%
VGG	68.56%	68.45%	0.11%
Resnet-50	73.11%	72.54%	0.57%
Resnet-152	75.18%	74.56%	0.61%

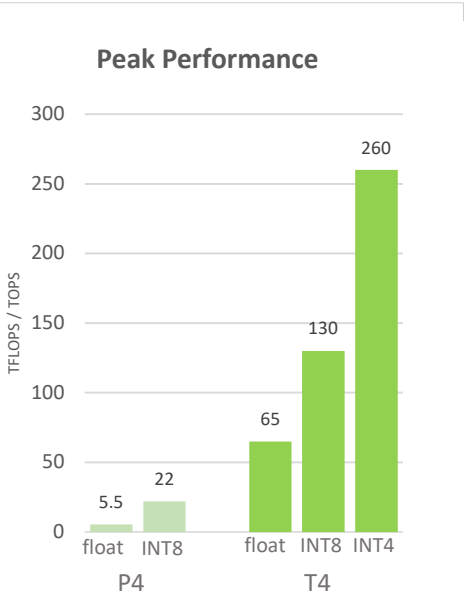
Precision calibration for INT8 inference:

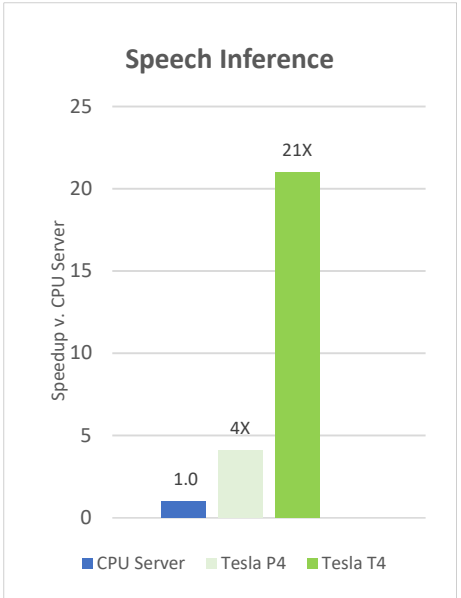
- Minimizes information loss between FP32 and INT8 inference on a calibration dataset
- Completely automatic



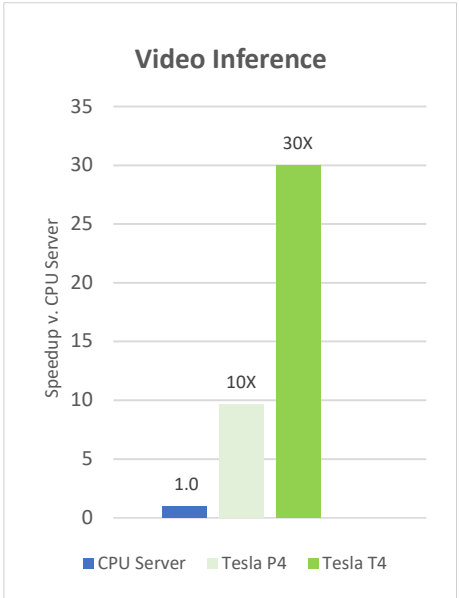
WORLD'S MOST PERFORMANT INFERENCE PLATFORM

Up To 36X Faster Than CPUs | Accelerates All AI Workloads

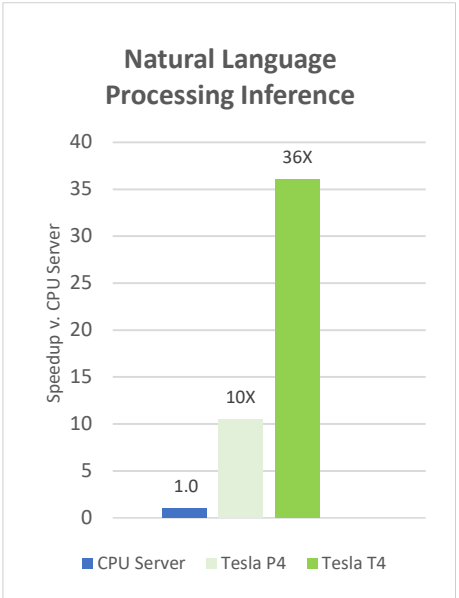




Speedup: 21X faster
DeepSpeech 2



Speedup: 30x faster
ResNet-50 (7ms latency limit)

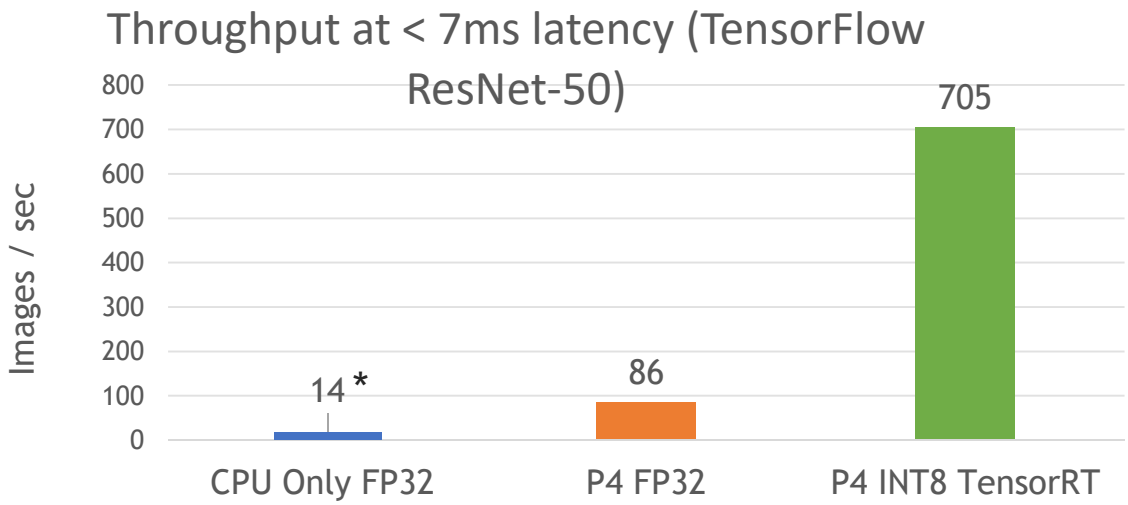


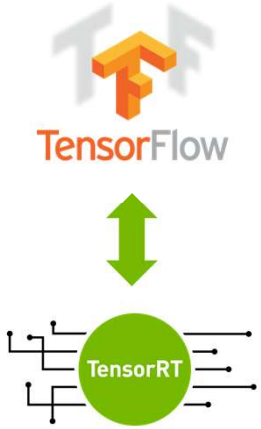
Speedup: 36x faster
GNMT

For all three graphs:
Dual-Socket Xeon Gold 6140 @ 3.6GHz with single GPU as shown | 19.01-py3 for T4 ResNet-50, 18.11-py3 | TensorRT 5.0 | CPU FP32, P4 & T4: INT8 | Batch Size = 128

TensorRT INTEGRATED WITH TensorFlow

8x faster Inference Than TensorFlow Only





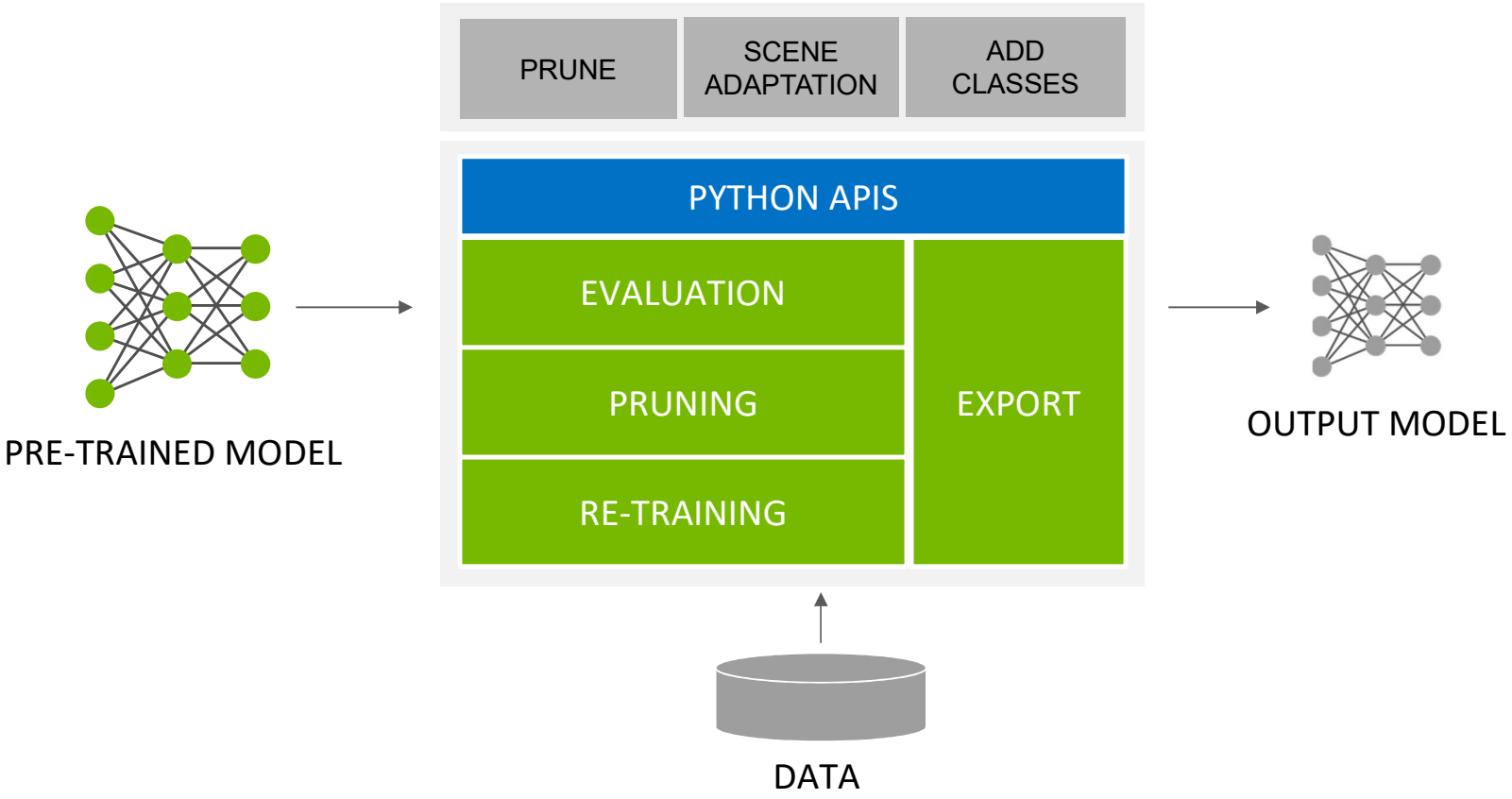
Available in TensorFlow 1.7 and above

https://github.com/tensorflow/tensorflow

* Min CPU latency measured was 70 ms. It is not < 7 ms.
CPU: Skylake Gold 6140, 2.5GHz, Ubuntu 16.04; 18 CPU threads.
Pascal P4; CUDA (384.111; v9.0.176); Batch size: CPU=1, TF_GPU=1 (latency 12 ms) , TF-TRT=4 w/ latency=6ms

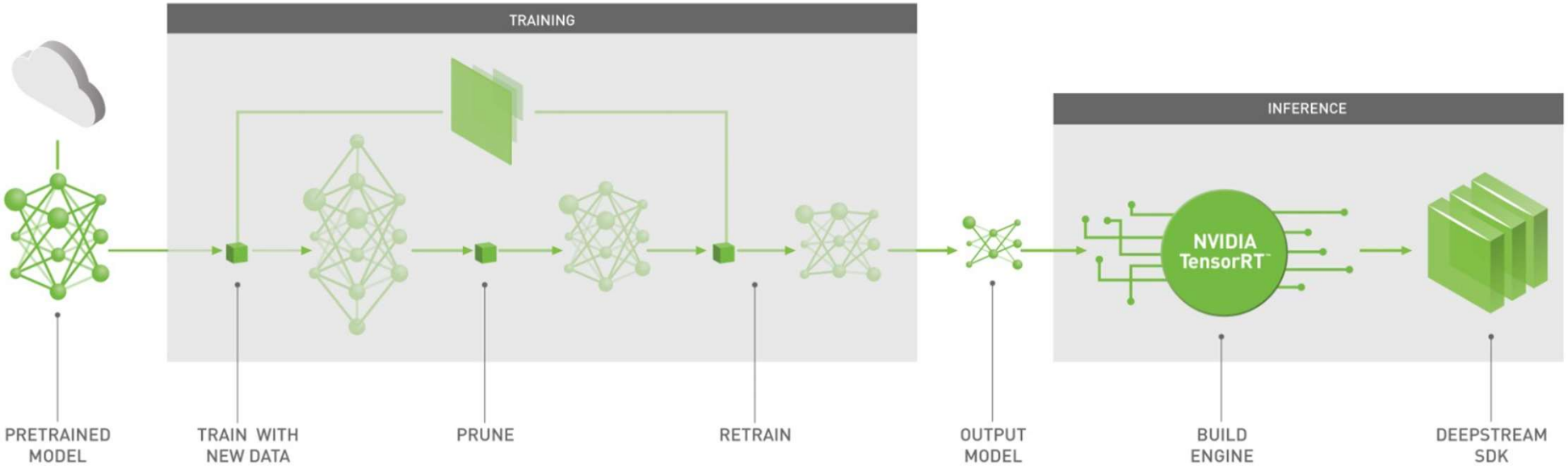

TRANSFER LEARNING
TOOLKIT

TRANSFER LEARNING TOOLKIT



End to End NVIDIA Deep Learning Workflow

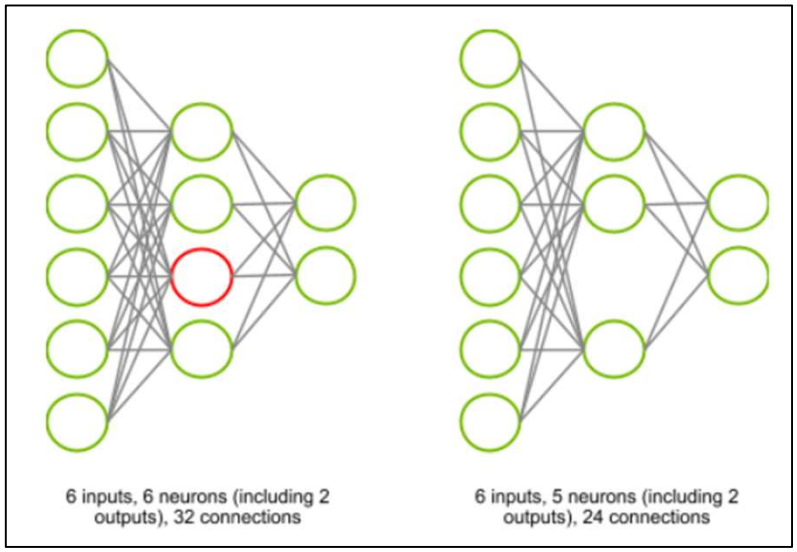
Pre-Trained model access from NGC * Training & adaptation * Applications ready to integrate with DeepStream



Accelerate time to market and save on compute resources!

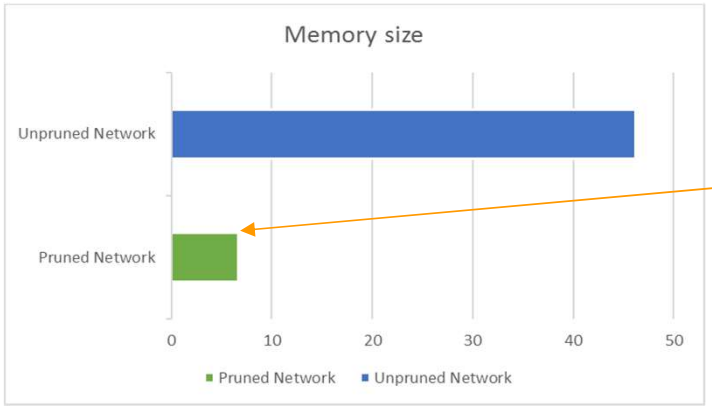
Pruning Models

1. Reduce model size and increase throughput
2. Incrementally retrain model after pruning to recover accuracy

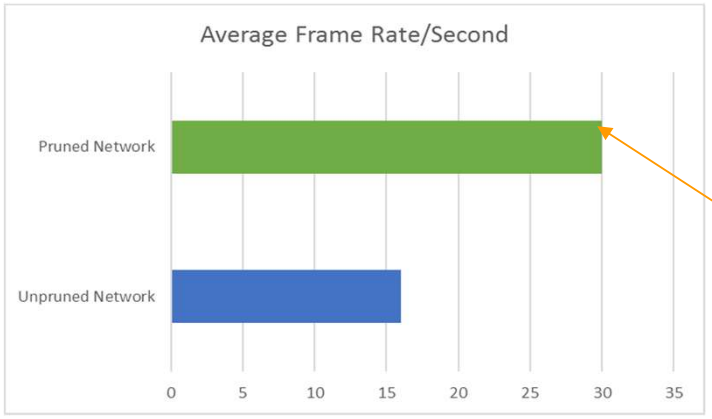


EXAMPLE

Network - ResNet 18 4-class (Car, Person, Bicycle, Road sign)
Memory size - 46.2 MB to 6.7 MB
FPS - 16 fps to 30 fps



6.5x
Model Size Reduction



>2x
Throughput Increase

FEATURES

Efficient Pre-trained Models

GPU-accelerated models trained on large scale public datasets.

Faster Inference with Model Pruning

Model pruning reduces size of the model resulting in faster inference

Training with Multiple GPUs

Re-training models, adding custom data for multi GPU training using an easy to use tool

Abstraction

Abstraction from having deep knowledge of frameworks, simple intuitive interface to the features

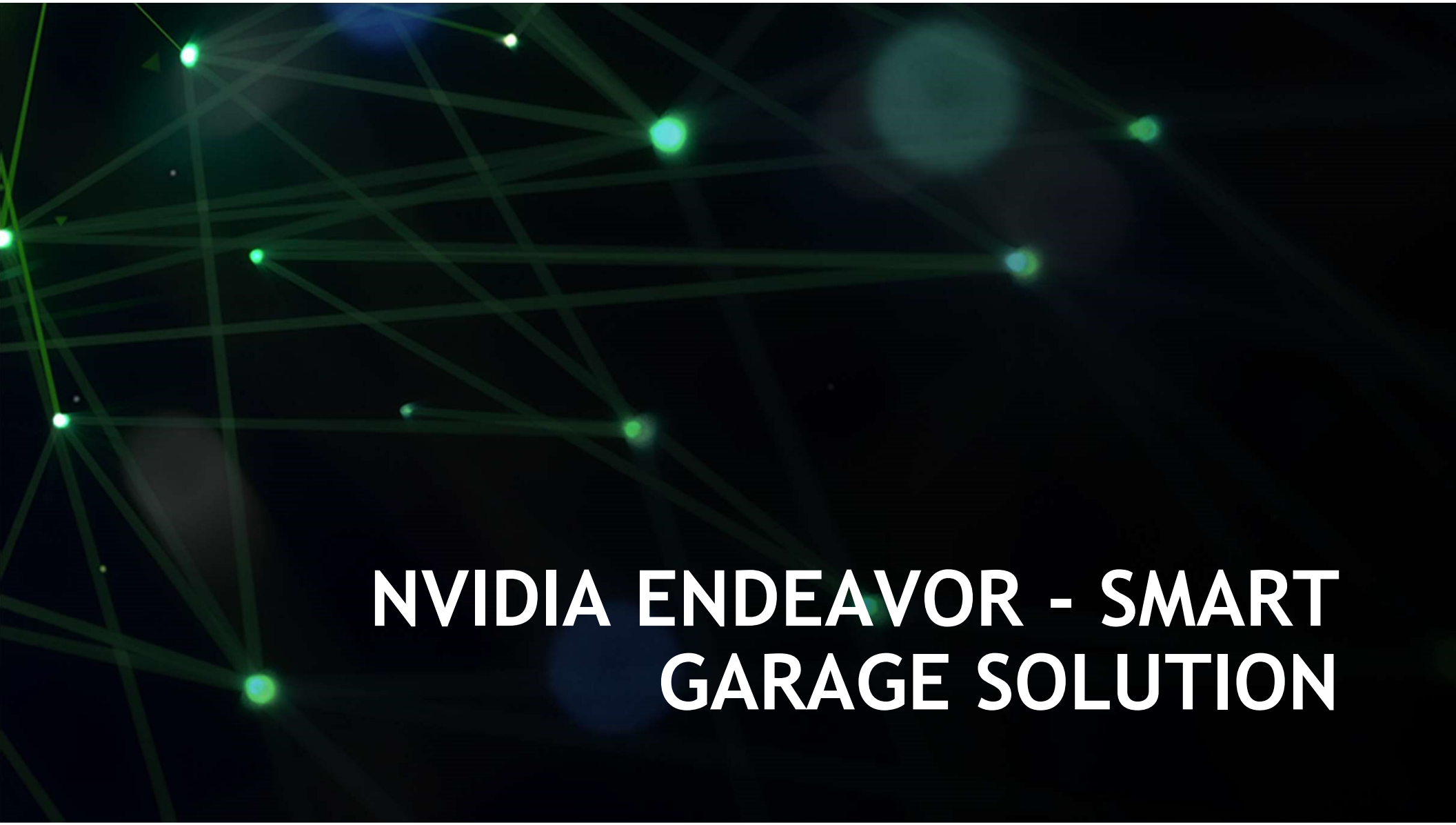
Containerization

Packaged in a container easily accessible from NVIDIA GPU Cloud website. All code dependencies are managed automatically

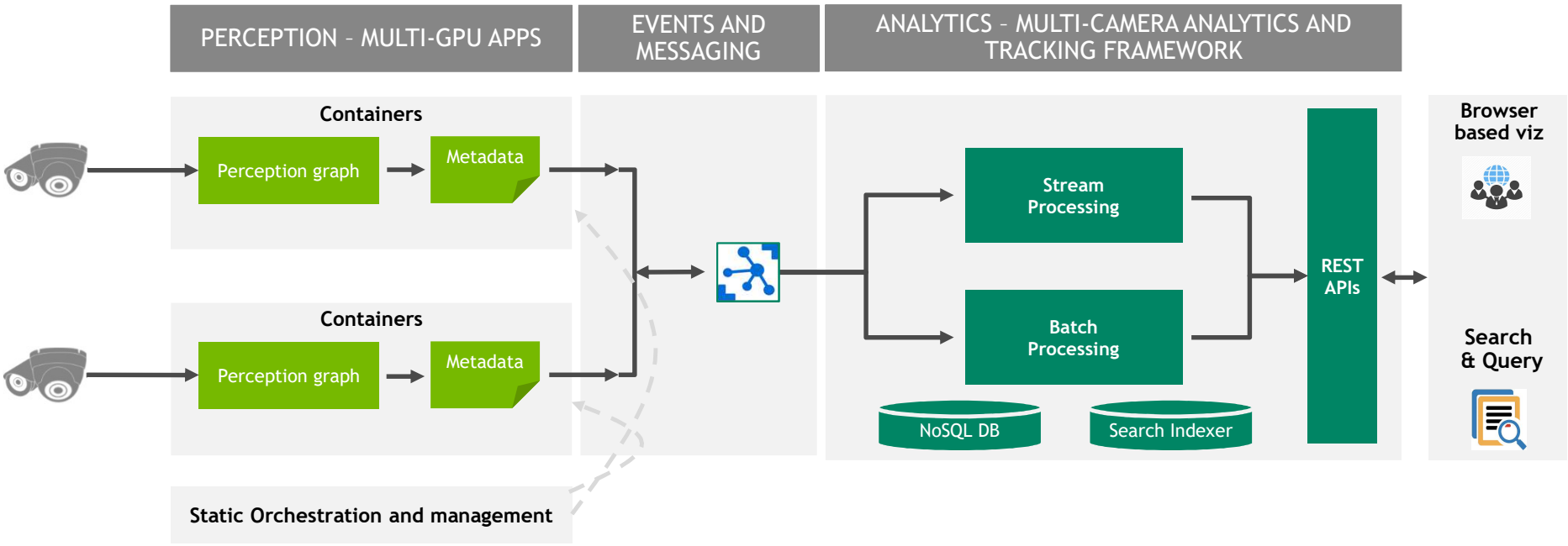
Integration

Models exported using TLT are easily consumable for inference with **Deep Stream SDK**

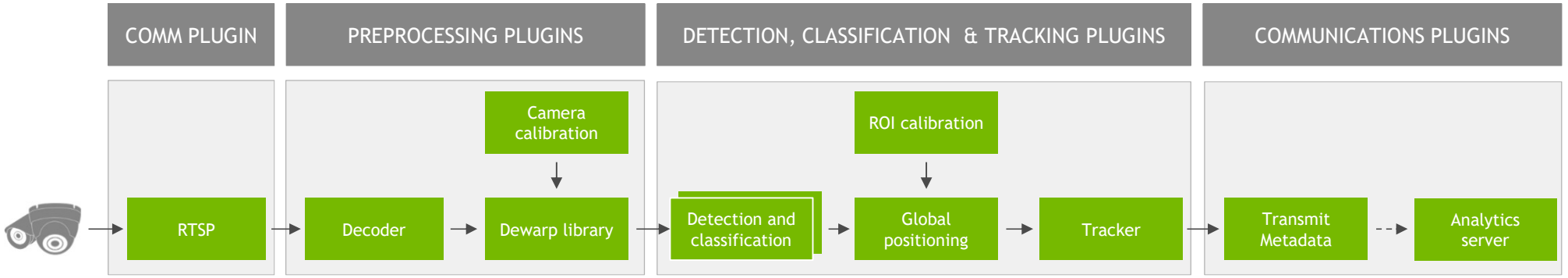

BUILD WITH DEEPSTREAM:
EXAMPLE APPLICATIONS


NVIDIA ENDEAVOR – SMART
GARAGE SOLUTION

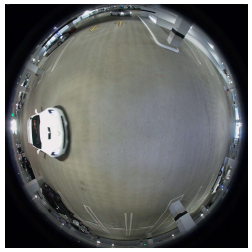
DEEPSTREAM 3.0 END-TO-END APPLICATION



PERCEPTION GRAPH

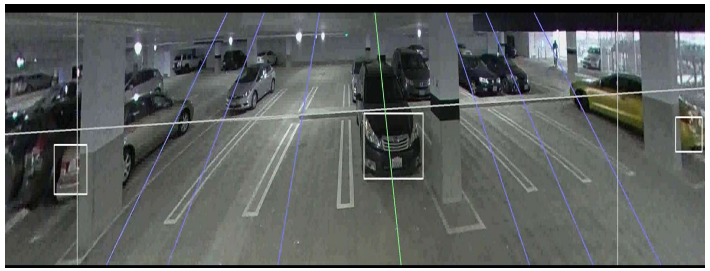


360d feeds

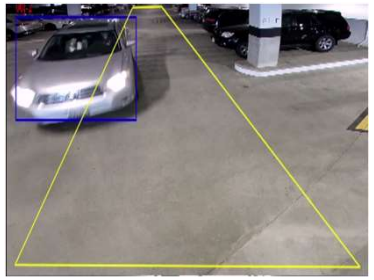


Dewarping

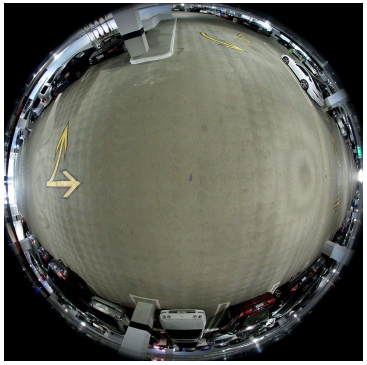
ROI: Lines



ROI: Polygon



ENABLING 360D CAMERA PROCESSING



NVWARP360
SDK

Equirectangular

Perspective

Cylindrical

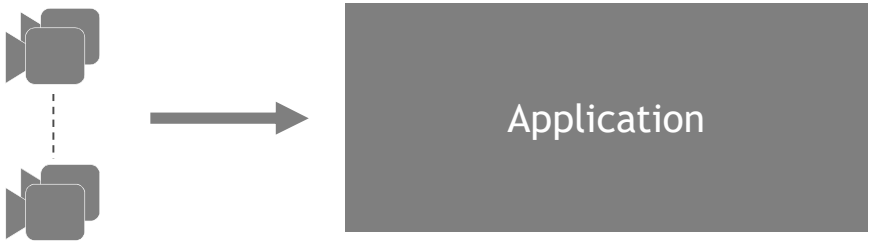
Rotated cylinder

Panini

Pushbroom

Tesla only

DYNAMIC STREAM MANAGEMENT

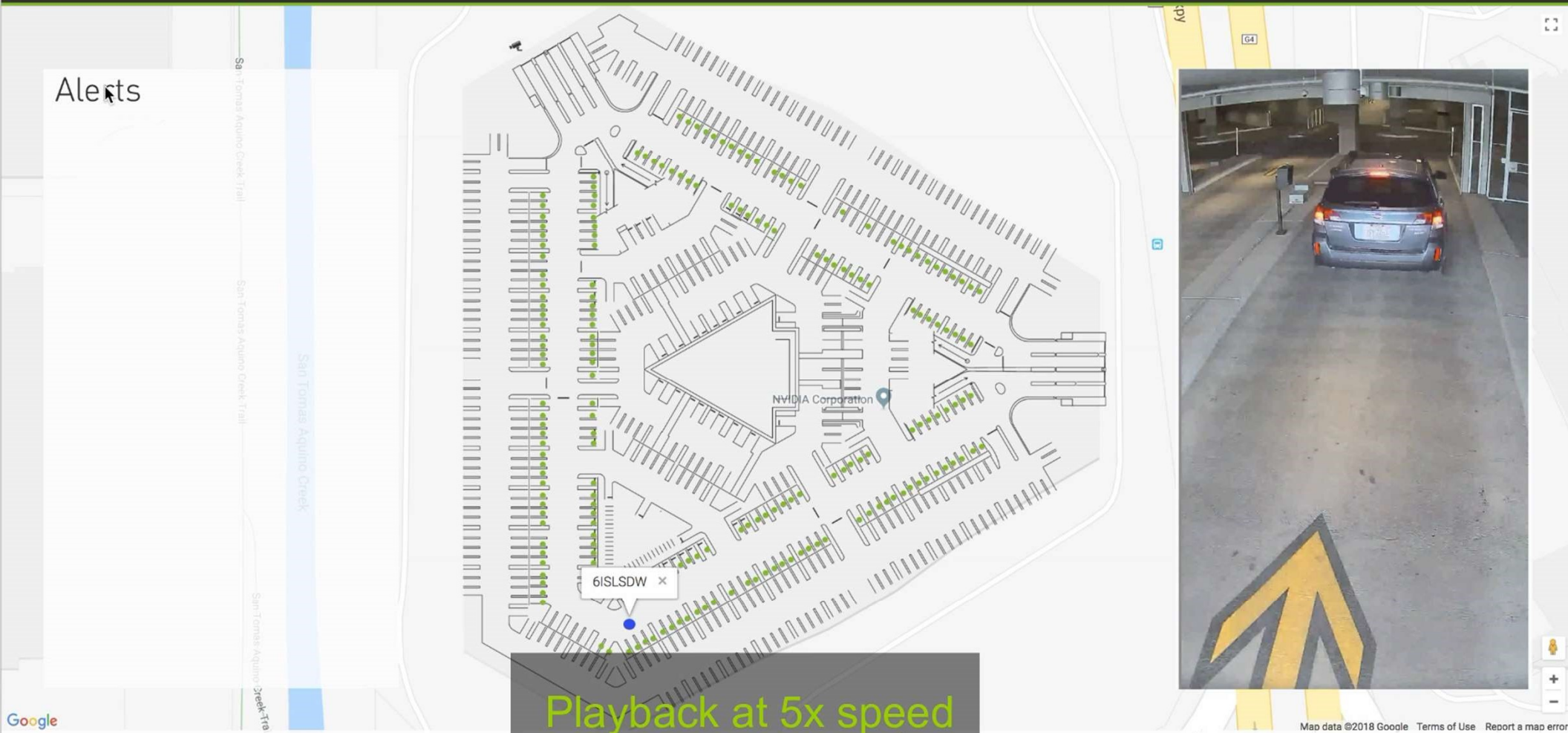


1. Add/ Remove camera streams
2. Change FPS
3. Change resolutions

NVIDIA

METROPOLIS

Alerts



Playback at 5x speed

Endeavor

1059 Free Spots

198 Occupied Spots

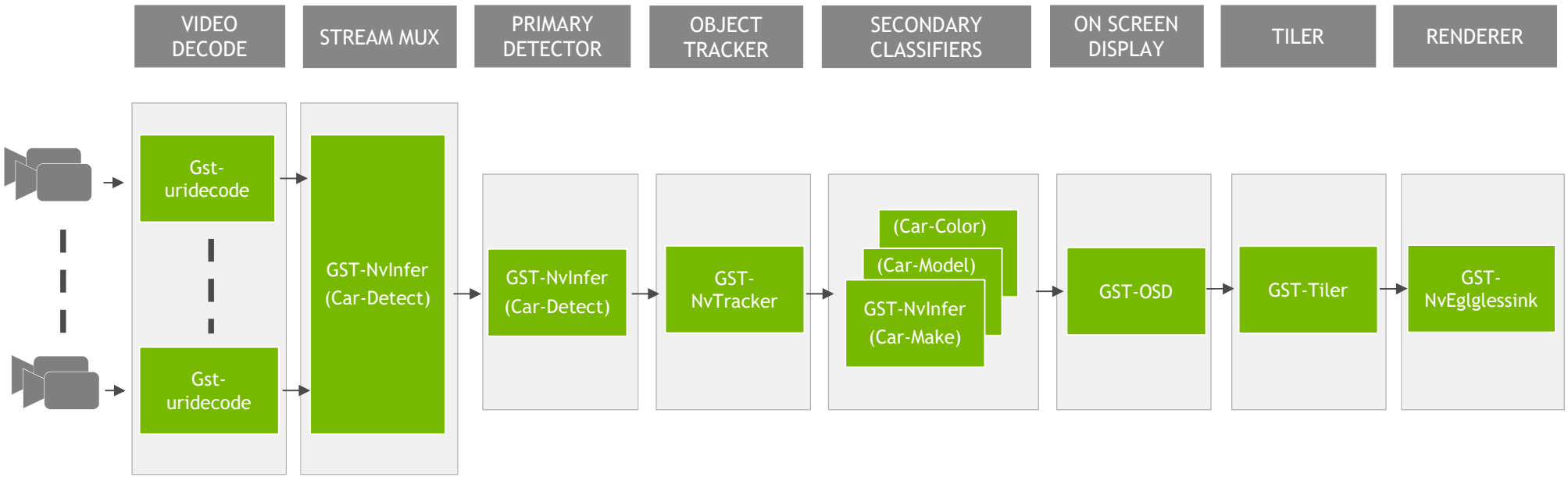
1/min Entry Flow Rate

0/min Exit Flow Rate

METROPOLIS | Ver: beta | User: Metropolis

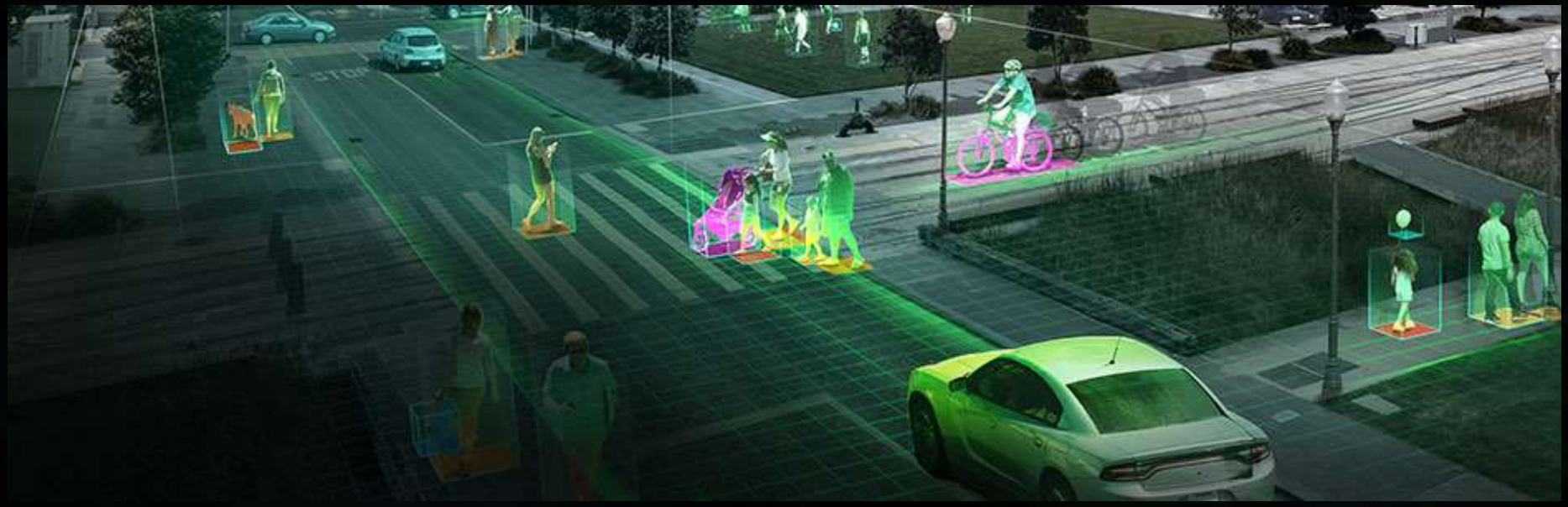

THIRTY STREAMS

MULTI-STREAM REFERENCE APPLICATION



REFERENCE APPLICATION VIDEO

START DEVELOPING WITH DEEPSTREAM



DEEPSTREAM . EXPLORE METROPOLIS . SUPPORT FORUMS

ONLINE RESOURCES

- NVIDIA DeepStream SDK
  - Product Page
- Blogs
  - Breaking the Boundaries of Intelligent Video Analytics with DeepStream SDK 3.0
  - Multi-Camera Large-Scale Intelligent Video
  - Using Calibration to Translate Video Data to the Real World
  - Accelerating Intelligent Video Analytics using Transfer Learning Toolkit
  - Accelerate Video Analytics Development with DeepStream SDK 2.0
- Webinars
  - Use Nvidia's DeepStream and Transfer Learning Toolkit to Deploy Streaming Analytics at Scale
  - Streamline Deep Learning for Video Analytics with DeepStream SDK 2.0

ONLINE RESOURCES

- Forums
  - Tesla Forum
  - Jetson Forum
- Software
  - DeepStream Container for Tesla and Sample Applications
  - JetPack (installer to flash your Jetson Developer Kit)
  - TensorRT
- GitHub Repositories
  - Reference Apps for Video Analytics using TensorRT 5 and DeepStream SDK 3.0
  - An Example of Using DeepStream SDK for Redaction
  - DeepStream 3.0 - 360 Degree Smart Parking Application
- Gstreamer Plugin and Application Development Guide
  - https://gstreamer.freedesktop.org/documentation/

Getting Started: Transfer Learning Toolkit

- Try Transfer Learning Toolkit. Access Open Beta today!
- Deploy end to end IVA solution with NVIDIA DeepStream 3.0. Download DeepStream 3.0
- Sign up for NVIDIA Developer Zone to access downloads, documentation and user tutorials
- Blogs:
  - What is Transfer Learning?
  - Pruning Models with Transfer Learning Toolkit
  - Accelerate IVA Applications with Transfer Learning Toolkit


NVIDIA