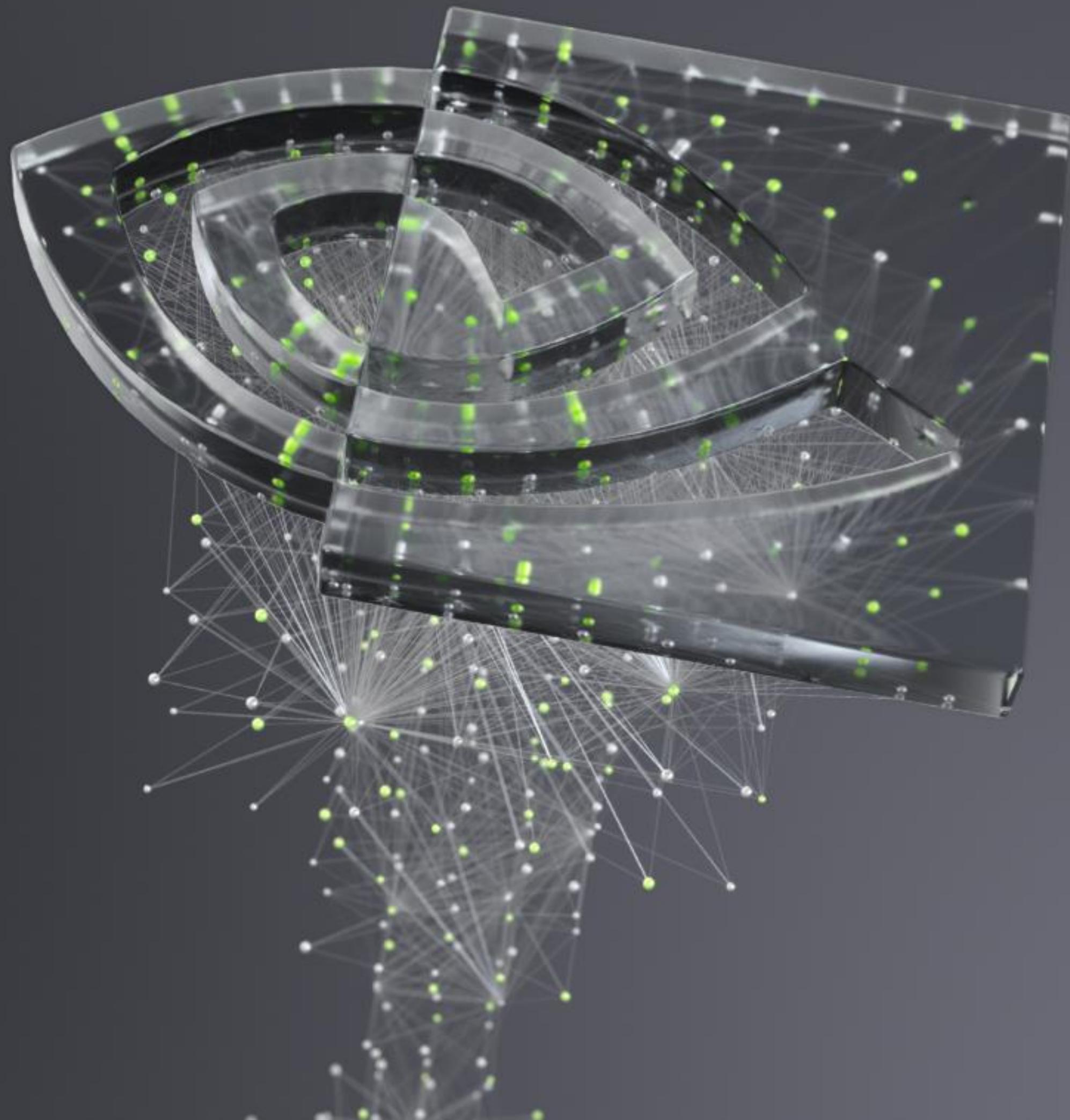
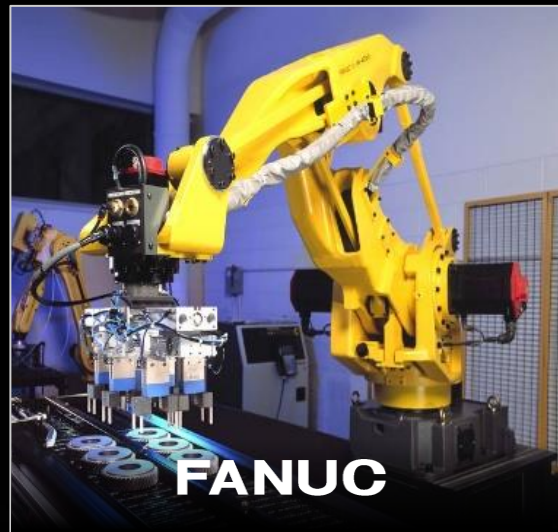




JETSON PLATFORM OVERVIEW



JETSON SUCCESS STORIES



Industrial



Aerospace/Defense



Construction



Agriculture



Healthcare



Smart City



Retail



Logistics



Delivery



Inspection



Service



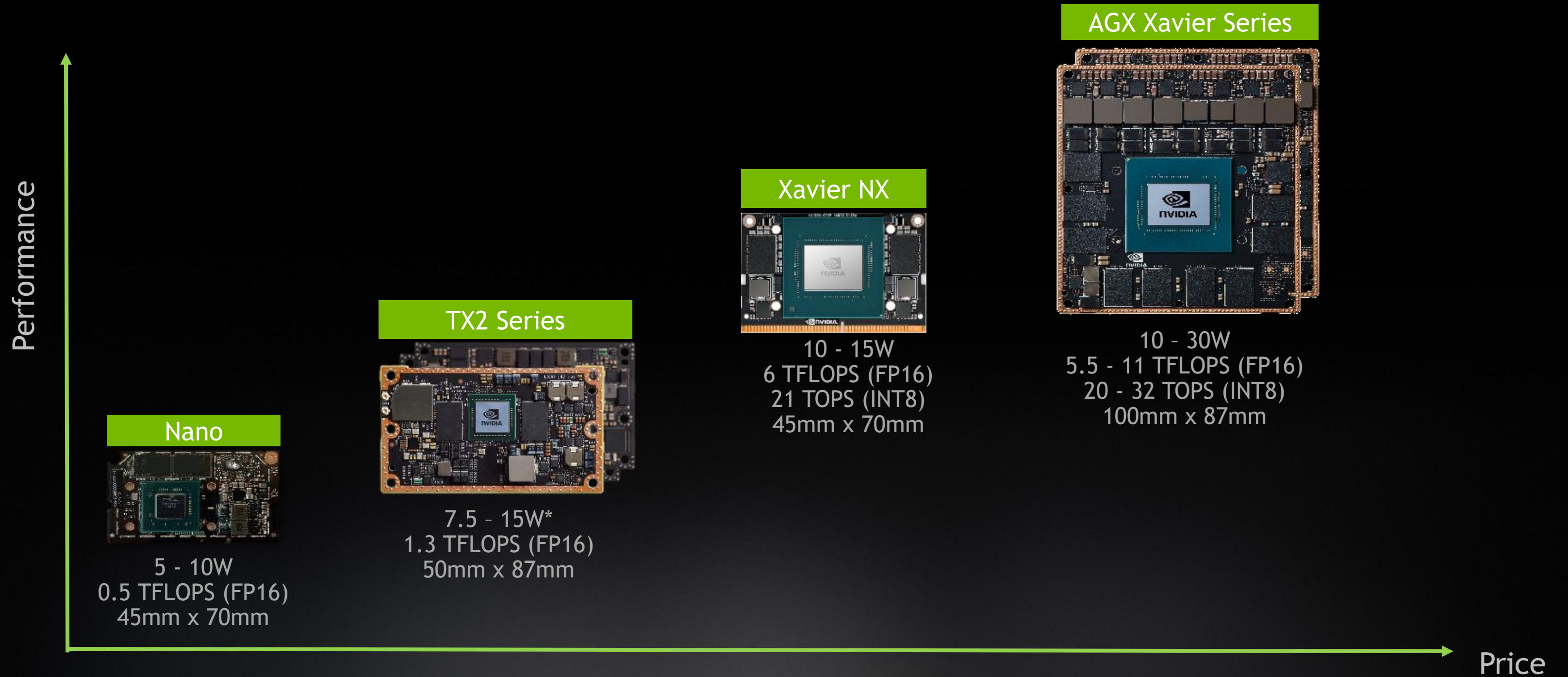
Collaboration



JETSON PRODUCT FAMILY OVERVIEW

THE JETSON FAMILY

From AI at the Edge to Autonomous Machines

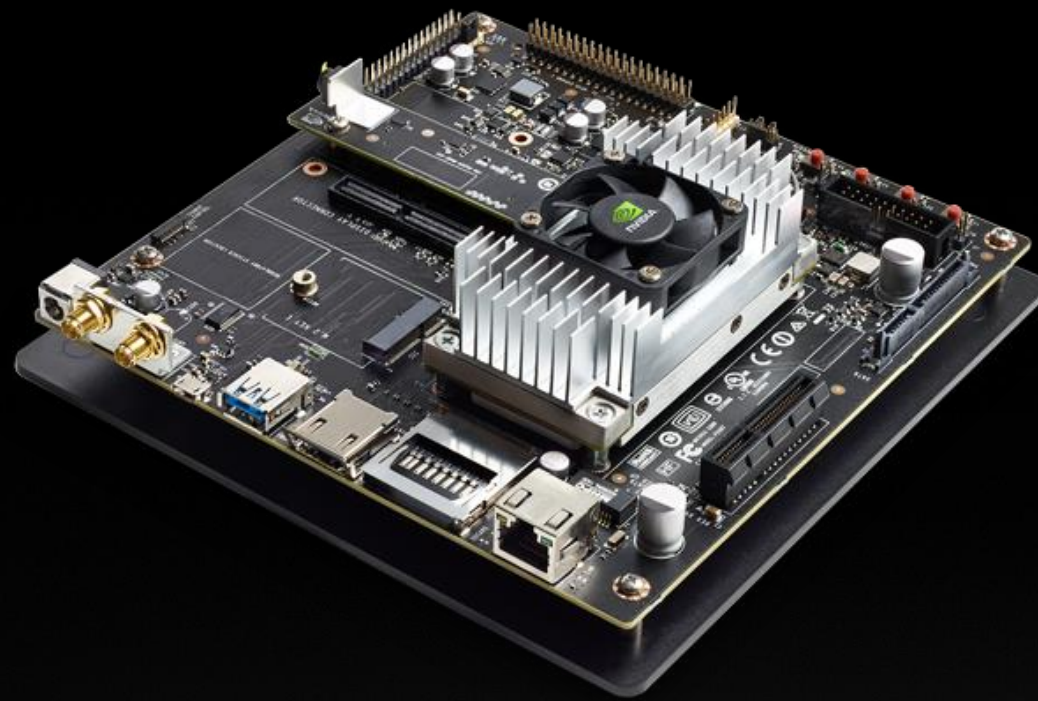


JETSON DEVELOPER KITS

For Developers, Engineers and Makers



JETSON NANO
5W | 10W
0.5 TFLOPS (FP16)
\$99



JETSON TX2
7.5W | 15W
1.3 TFLOPS (FP16)
\$399 (\$299 EDU)



JETSON XAVIER NX
10W | 15W
7 TFLOPS (FP16) | 21 TOPS (INT8)
\$399

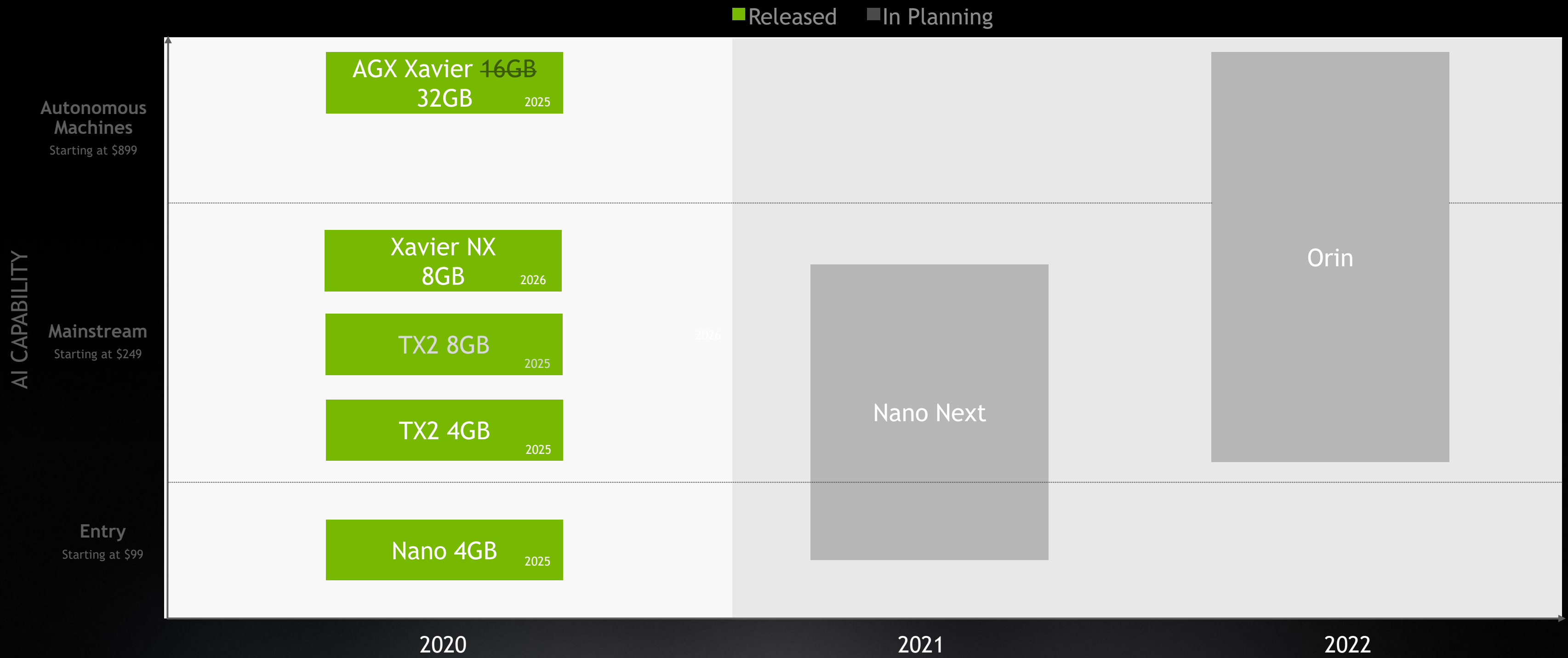


JETSON AGX XAVIER
10 | 15W | 30W
11 TFLOPS (FP16) | 32 TOPS (INT8)
\$699

Multiple developer kits - Same software

Full specs at developer.nvidia.com/jetson

JETSON MODULES — COMMERCIAL ROADMAP

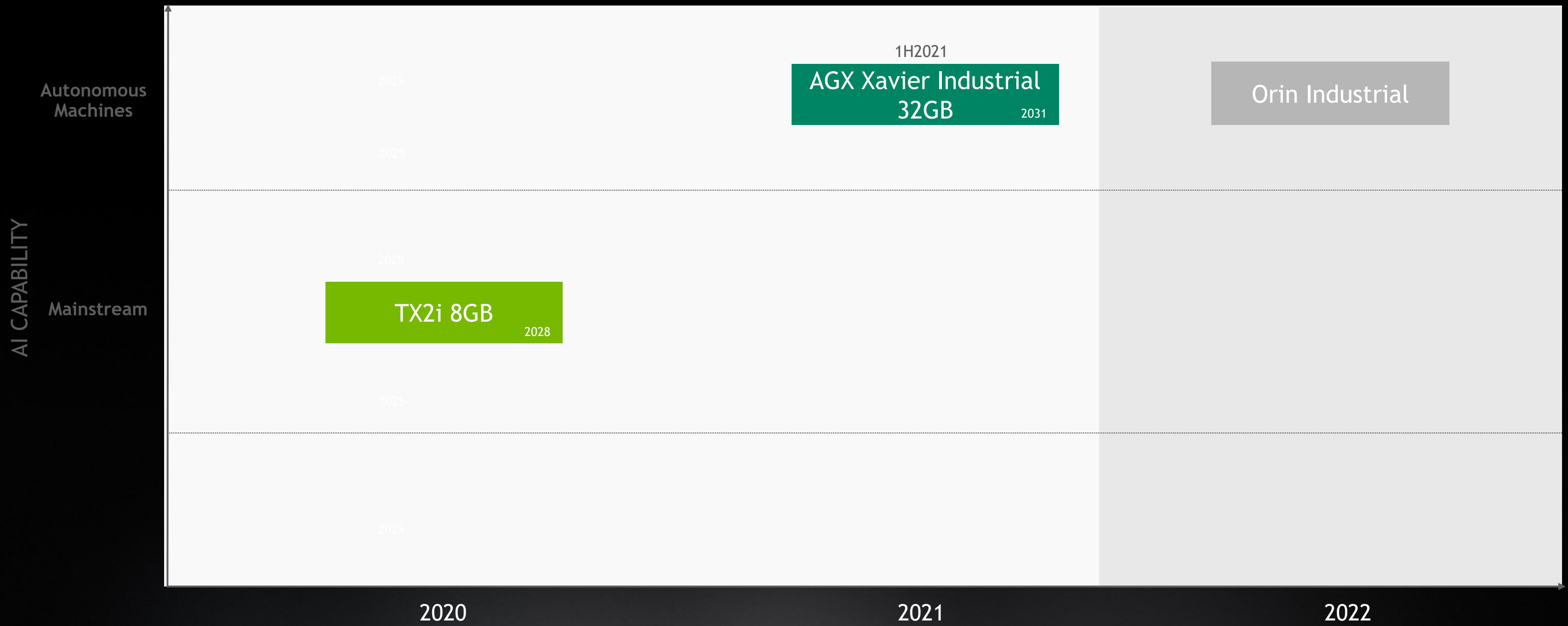


* TX2 8GB will ship till 2025 but new designs recommend transition to NX.

The year in each box indicates supply availability at least / until. Lifecycle might be extended further, contact your partner sales manager.
Operating life of Jetson is 5 years 24x7 for commercial production modules. Products in development and planning are subject to change.

JETSON MODULES — INDUSTRIAL ROADMAP

■ Released ■ In Development ■ In Planning



*The year in each box indicates supply availability at least / until. Lifecycle might be extended further, contact your partner sales manager.
 Operating life of Jetson industrial version is 10 years 24x7.
 Products in development and planning are subject to change.*

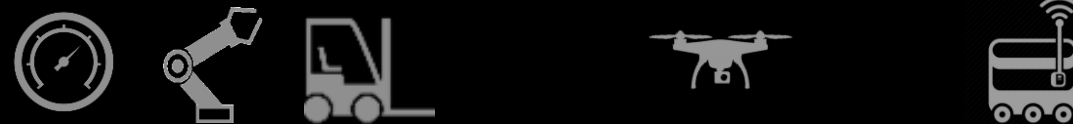
JETSON SOFTWARE

for AI Edge Devices

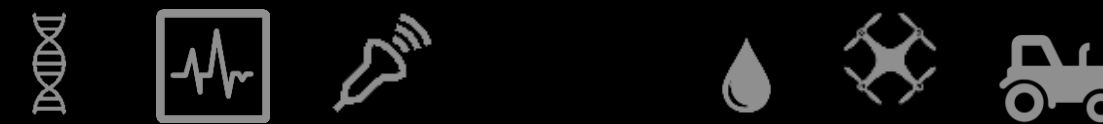
City



Logistics



Agriculture



Ecosystem

AI Software and Services

Ecosystem

Machine Vision Cameras & Sensors

Ecosystem

System Software & Developer Tools

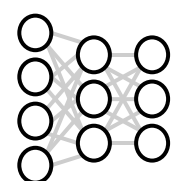
DeepStream SDK

Isaac SDK

JetPack SDK

CUDA-X

Deep Learning



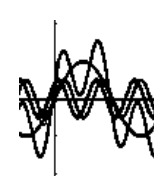
TensorRT
cuDNN

Multimedia



libargus
Video API

Accelerated Computing



cuBLAS
cuFFT

Computer Vision



VPI
VisionWorks
OpenCV

Sensors



Drivers
Ecosystem

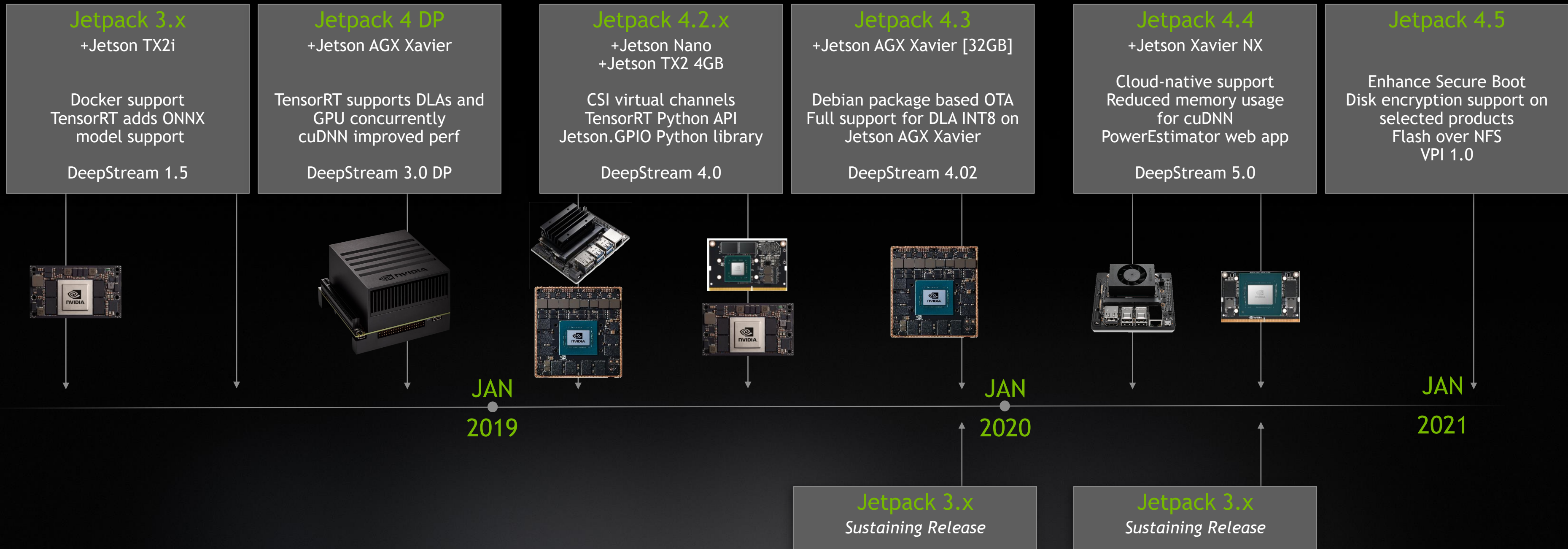
Developer Tools

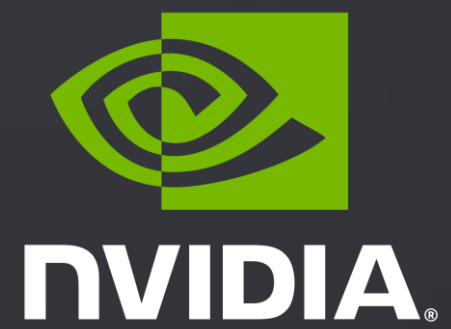
CUDA • Linux • RTOS

Jetson

CONTINUOUS SOFTWARE INVESTMENT

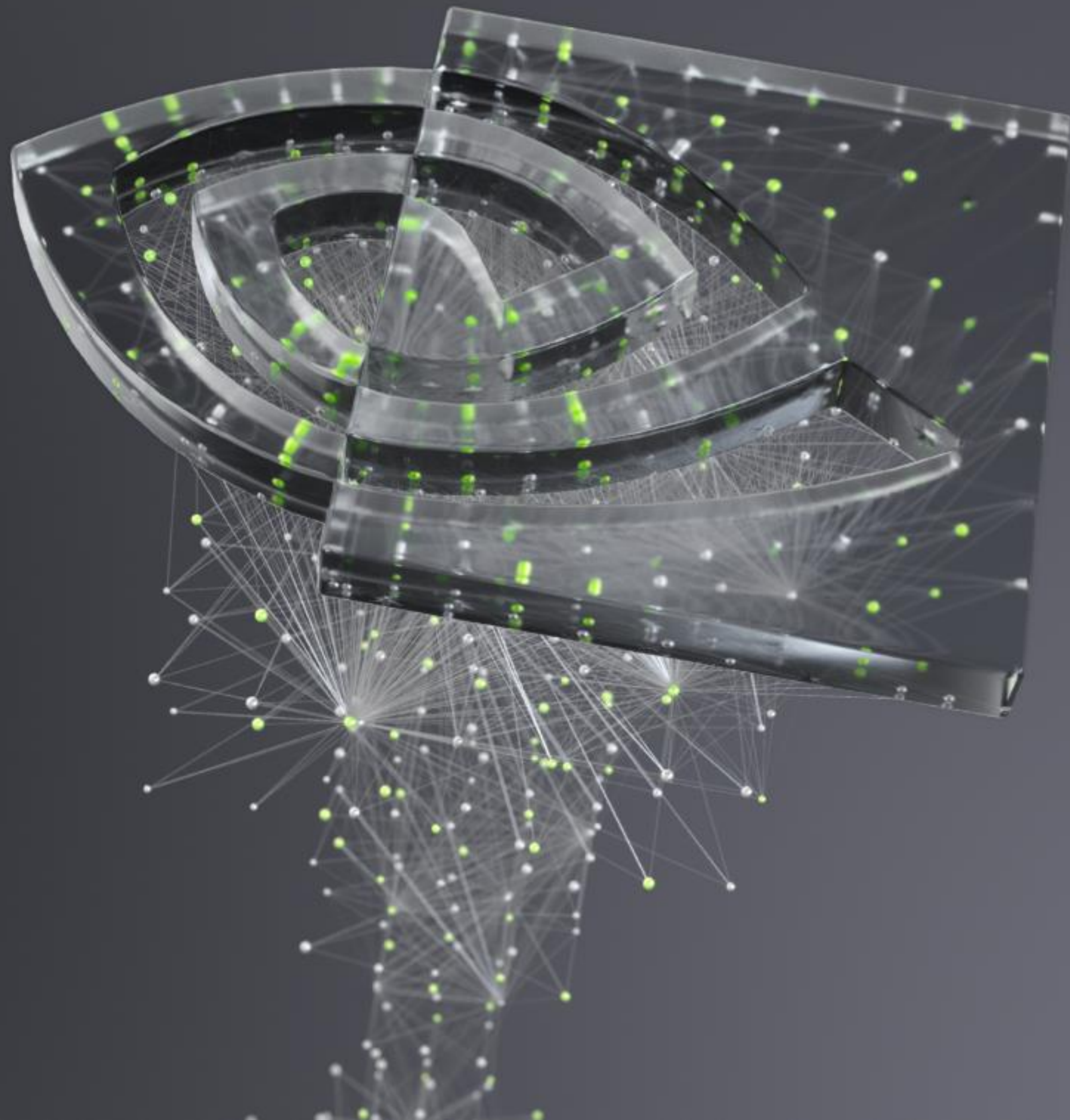
Scalable Software Defined Platform





DEEPSTREAM

March 2020



INTELLIGENT VIDEO ANALYTICS (IVA) FOR EFFICIENCY AND SAFETY

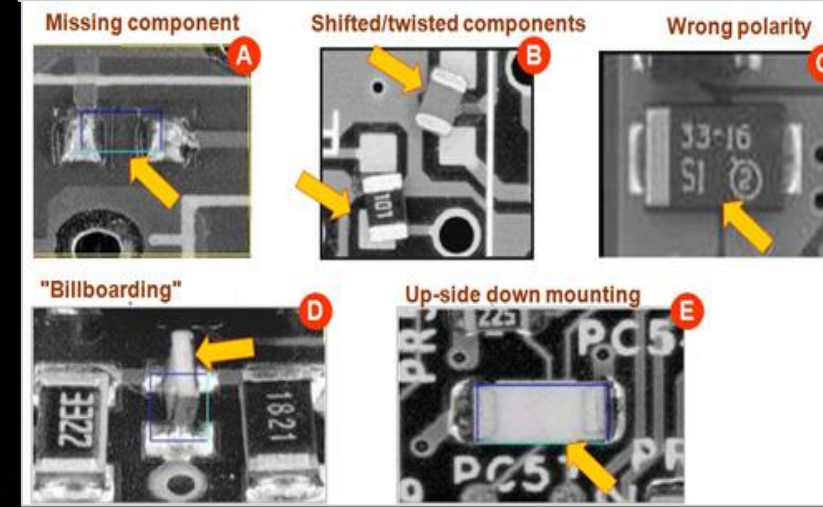
Access Control



Public Transit



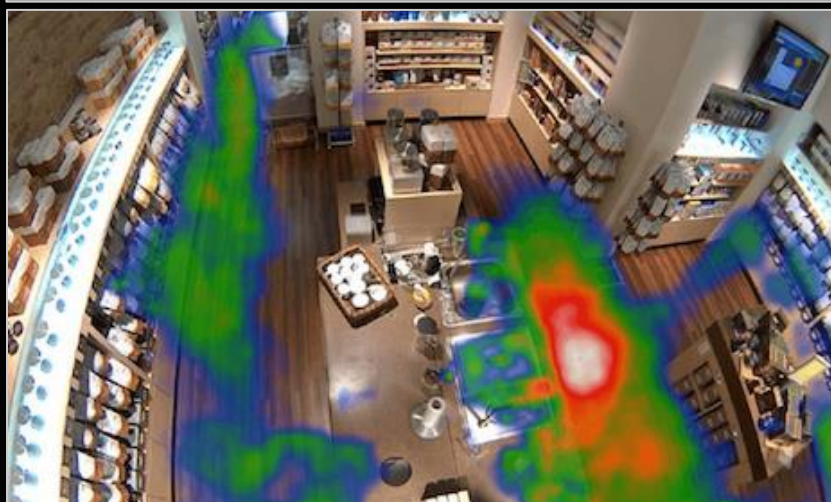
Industrial Inspection



Traffic Engineering



Retail Analytics



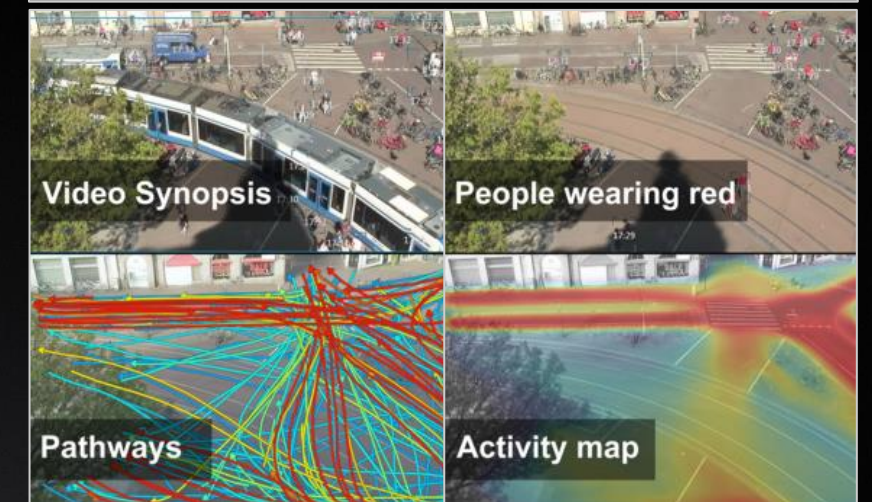
Logistics



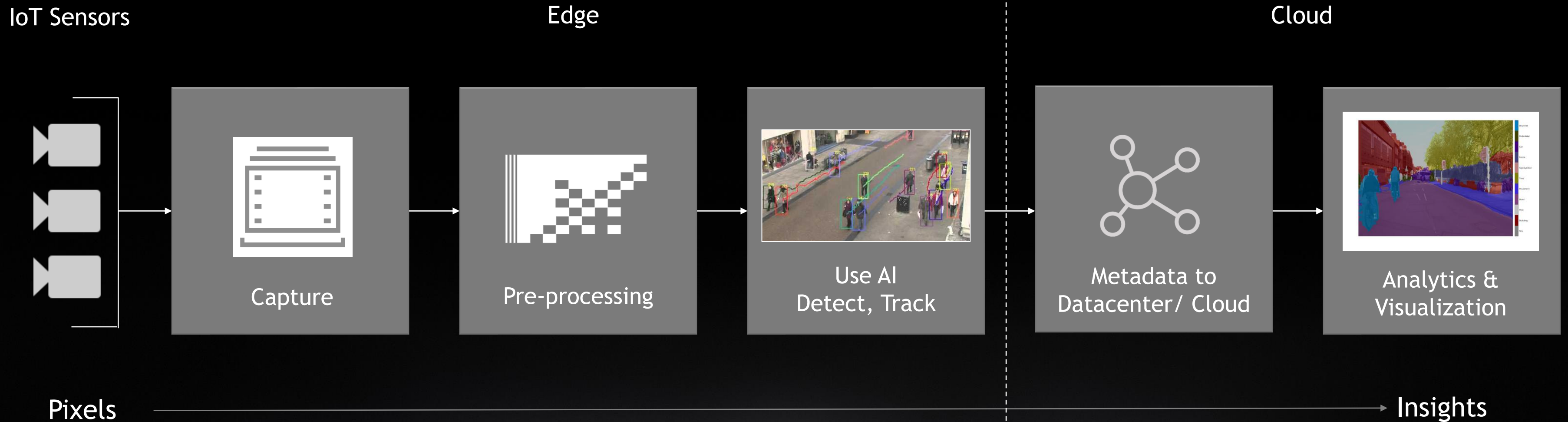
Critical Infrastructure



Public Safety



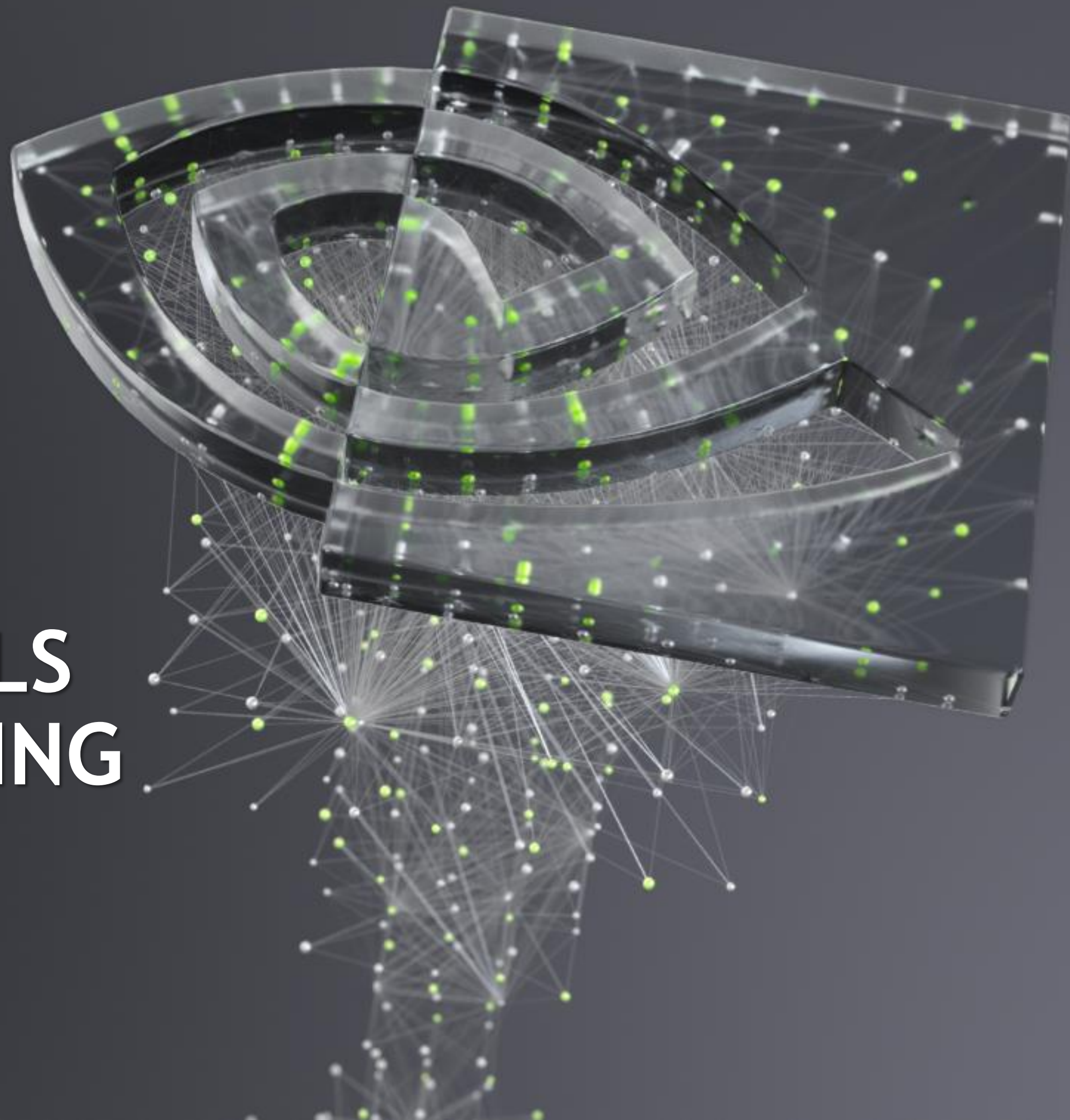
IVA APPLICATION WORKFLOW





nvidia

PRE-TRAINED MODELS & TRANSFER LEARNING TOOLKIT



PRODUCTION READY PRE-TRAINED VISION AI MODELS



People Detection



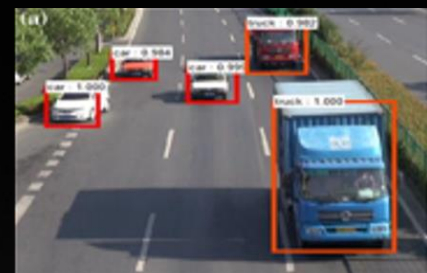
Face Detect IR



DashCamNet



TrafficCamNet



Vehicle Type net



Vehicle Make net

Free AI models for **Smart City, Public Safety, Retail, Healthcare & Robotics**

Over **80%** accuracy

Optimized for **High Throughput**

Deploy with turnkey inference examples

Transfer Learn with your dataset

Check out the models on [NGC](#)

MODEL CARDS READILY AVAILABLE ON NGC

NVIDIA NGC | CATALOG
Welcome Guest

Catalog: Models / Models: nvidia:tlt_peoplenet

PeopleNet

Publisher	Application	Version	Modified
NVIDIA	Object Detection	unpruned_v2.0	September 24, 2020

Framework	Model Format	Precision	GPU Model
Transfer Learning Toolkit	TLT	FP32	V100

Description
Highly accurate purpose-built 3 class object detection network to detect people in an image.

Labels

CV IVA Object Detection TLT Computer Vision Deep Learning Healthcare Intelligent Video Analytics Retail Robotics Smart Cities Training Transf

Widget Model CLI Command

```
$ wget --content-disposition https://api.ngc.nvidia.com/v2/models/nvidia/tlt_peoplenet/versions/unpruned_v2.0/zip -O tlt_peoplenet_unpruned_v2.0.zip
```

Overview Version History File Browser Related Collections

PeopleNet Model Card

Model Overview

The models described in this card detects one or more physical objects from three categories within an image and returns a box around each object, as well as the bounding-box coordinates and confidence value per output class.

Model Architecture

These models are based on NVIDIA DetectNet_v2 detector with ResNet34 and ResNet18 as feature extractors. This architecture, also known as GridBox object detection, divides an input image into a grid which predicts four normalized bounding-box parameters (xc, yc, w, h) and confidence value per output class.

The raw normalized bounding-box and confidence detections needs to be post-processed by a clustering algorithm such as DBSCAN or NMS to produce final bounding-box coordinates and category labels.

Training Data

PeopleNet v1.0 model was trained on a proprietary dataset with more than 5 million objects for person class. The training dataset consists of a mix of camera heights, crowd-density, and field-of-view (FOV). Approximately half of the training data consisted of images captured in an indoor office environment. For this case, the camera is typically set up at approximately 10 feet height, 45-degree angle and has close field-of-view. This content was chosen to improve accuracy of the models for convenience-store retail analytics use-case.

Environment	Images	Persons	Bags	Faces
5ft Indoor	108,692	1,060,960	235,992	235,992
5ft Outdoor	206,912	166,8250	657,162	657,162
10ft Indoor (Office close FOV)	413,278	4,577,870	960,968	1,664,939
10ft Outdoor	18,321	178,817	56,890	56,097
20ft Indoor	104,972	1,079,550	397,793	337,083
20ft Outdoor	24,783	59,623	46,173	27,840
Total	876,958	8,626,070	2,354,978	2,379,113

Training Data Ground-truth Labeling Guidelines

The training dataset is created by labeling ground-truth bounding-boxes and categories by human labelers. Following guidelines were used while labeling the training data for NVIDIA PeopleNet model. If you are looking to re-train with your own data, please refer to the PeopleNet project labeling guide.

- All objects that fall under person class are labeled with the appropriate bounding-box and category.
- If a person is carrying an object, the object is labeled with the appropriate bounding-box and category.
- Occlusion: For partially occluded objects, the bounding-box and category are marked as partially occluded.
- Occlusion for person class: If a person's head and shoulders are not visible, the object is marked as truncated.
- Truncation: For an object of person class, if the head and shoulders are not visible, the object is marked as truncated.
- Truncation for person class: If the head and shoulders are not visible, the object is marked as truncated.
- Each frame is not required to have a person.

Evaluation Data

The inference performance of PeopleNet v1.0 model was measured against 42000 proprietary images across a variety of environments. The frames are high resolution images 1920x1080 pixels before passing to the PeopleNet detection model.

Methodology and KPI

The true positives, false positives, false negatives are calculated using intersection-over-union (IOU) criterion greater than 0.5. The KPI for the evaluation data are reported in the table below based on precision, recall and accuracy.

Model	ResNet 34			ResNet 18		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
5ft	86.15	90.36	78.90	85.6	88.9	77.31
10ft	91.54	82.20	76.40	92.5	81.06	76.09
20ft	92.09	89.22	85.60	91.84	89.76	83.74
Office use-case	94.66	89.15	84.89	95.65	87.05	83.74

Real-time Inference Performance

The inference is run on the provided pruned models at INT8 precision. On the Jetson Nano FP16 precision is used. The inference performance is run using `trtexec` on Jetson Nano. AGX Xavier inference performance is run using `trtexec` on Jetson AGX Xavier. The Jetson devices are used for inference performance. The inference performance data might slightly vary depending on the hardware configuration.

Limitations

Very Small Objects
NVIDIA PeopleNet model were trained to detect objects larger than 10x10 pixels. Therefore it may not be able to detect very small objects.

Occluded Objects
When objects are occluded or truncated such that less than 20% of the object is visible, they may not be detected. However if the person's head and/or shoulders are not visible, the object might not be detected.

Dark-lighting, Monochrome or Infrared Camera Images
The PeopleNet model were trained on RGB images in good lighting conditions. Therefore, images captured in dark, monochrome or infrared camera images may not yield good results.

Warped and Blurry Images
The PeopleNet models were not trained on fish-eye lense cameras or moving cameras. Therefore, the models may not yield good results on warped and blurry images.

Face and Bag class
Although bag and face class are included in the model, the accuracy of these classes will be much lower than person class.

TRAINING DATASET

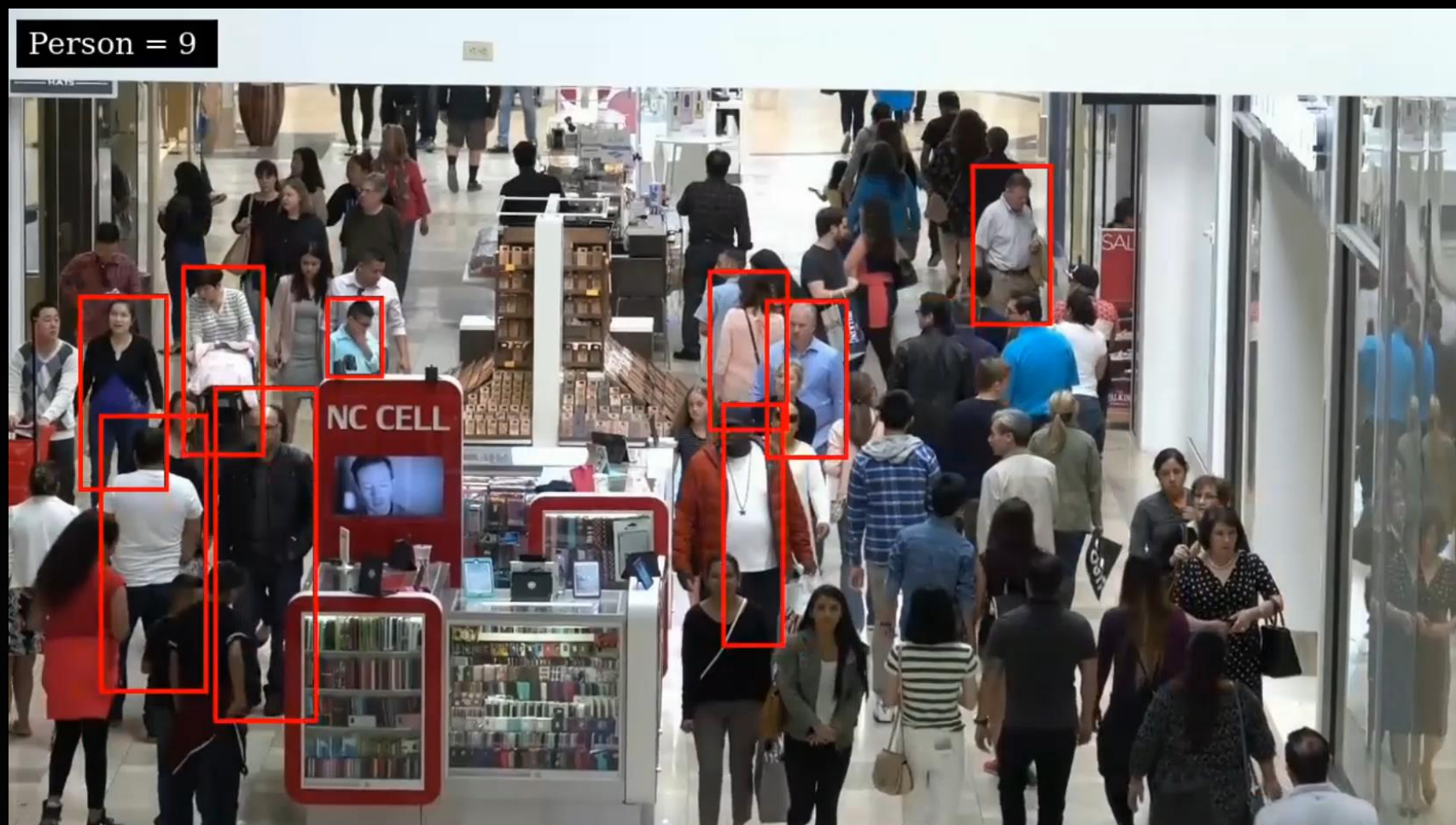
VALIDATION DATASET

CAMERA GUIDELINES

LIMITATIONS

FASTEST TIME TO MARKET FOR PEOPLE DETECTION

YOLOV4 model



Lots of data required to improve accuracy
Large training time

PeopleNet model

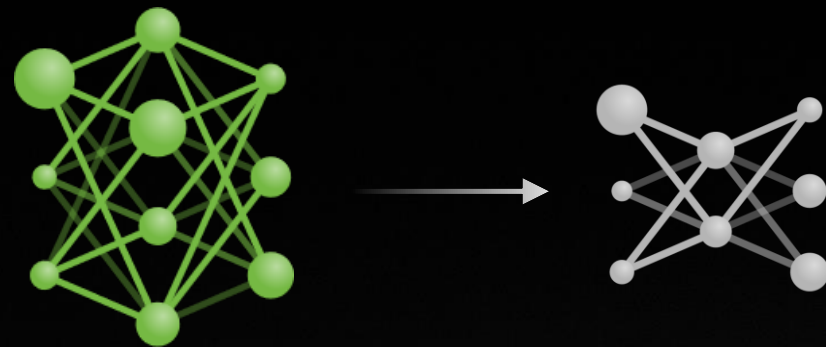


Apply minimal fine-tuning or deploy as is

WHAT IS TRANSFER LEARNING?

“**Transfer Learning** is a process of transferring learned features from one model to another”.

With Transfer Learning, you start with a pre-trained model and a small dataset, fine-tuning the model to learn from to your dataset. The pre-trained weights provide a better starting point to train on your dataset instead of starting from random weights.



NVIDIA **Transfer Learning Toolkit (TLT)** brings the transfer learning capability in a simplified training toolkit where you start with from a rich library of NVIDIA provided **pre-trained models** to build you custom AI network in 1/10th the effort and time versus starting from scratch.



Less Data Required to Train Accurately



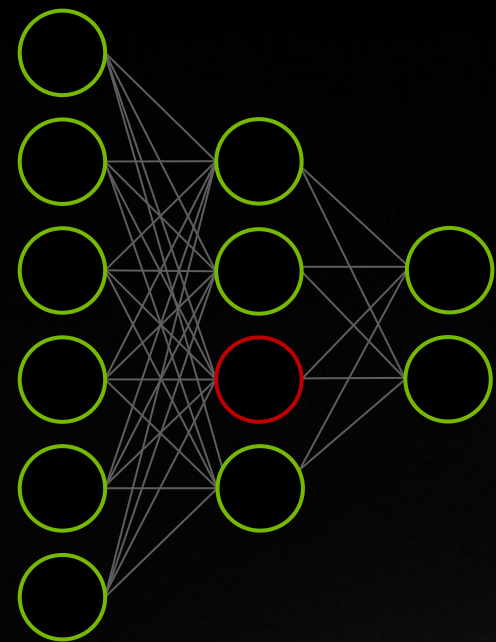
Reduce Training Time and Cost

<https://blogs.nvidia.com/blog/2019/02/07/what-is-transfer-learning/>

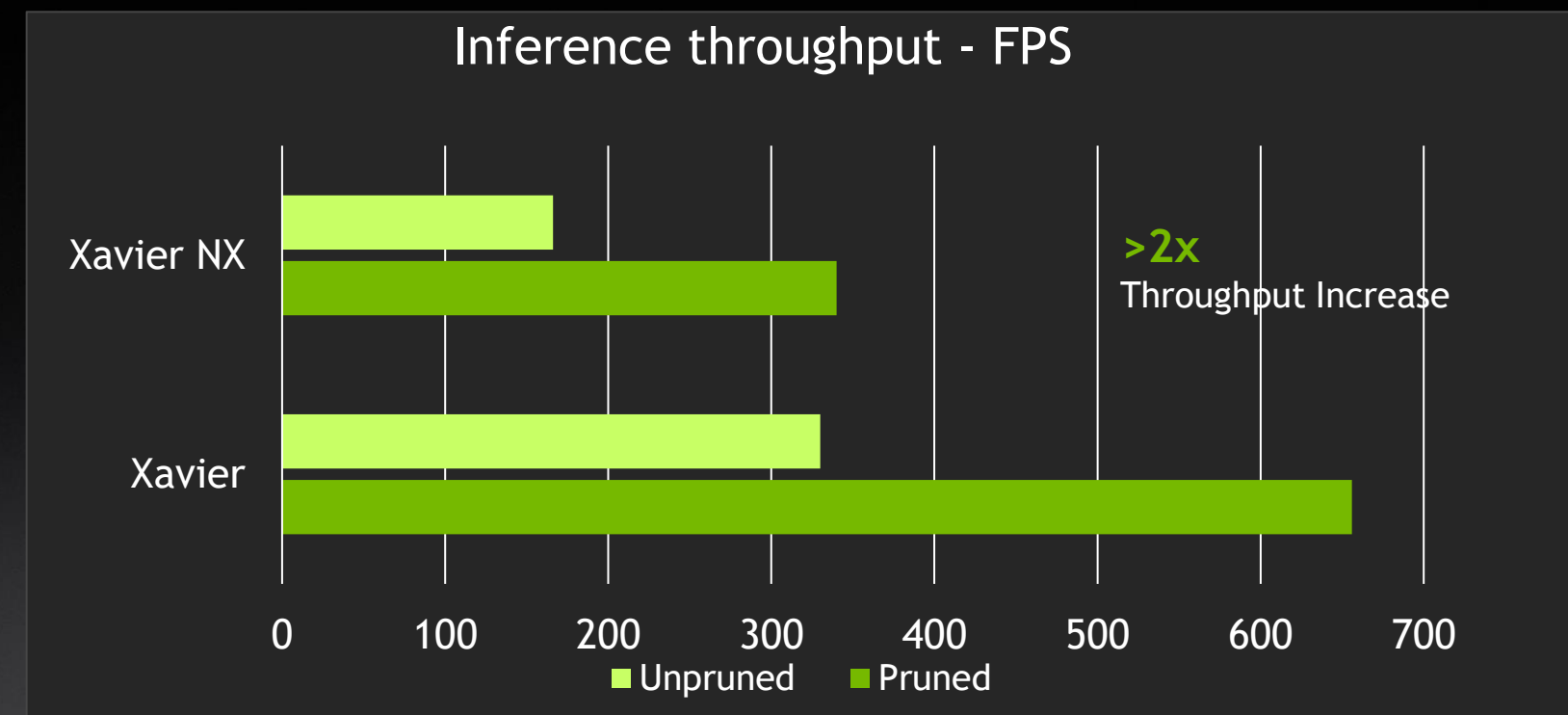
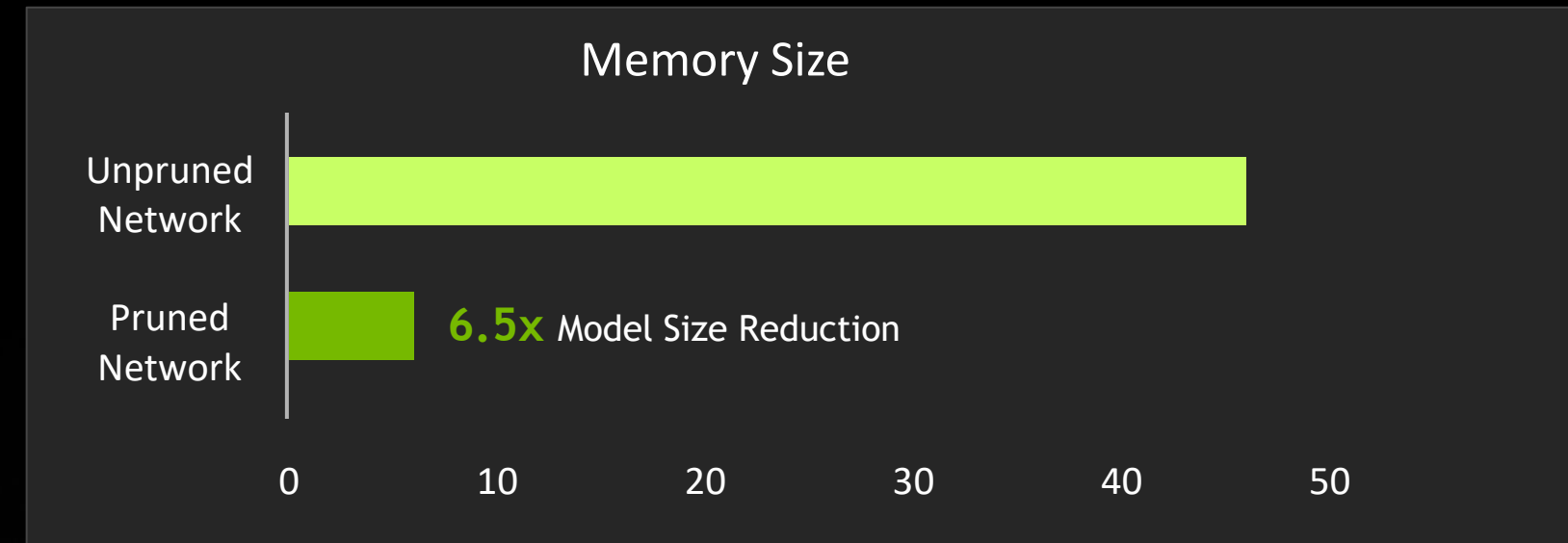
MODEL PRUNING

2 Step Process

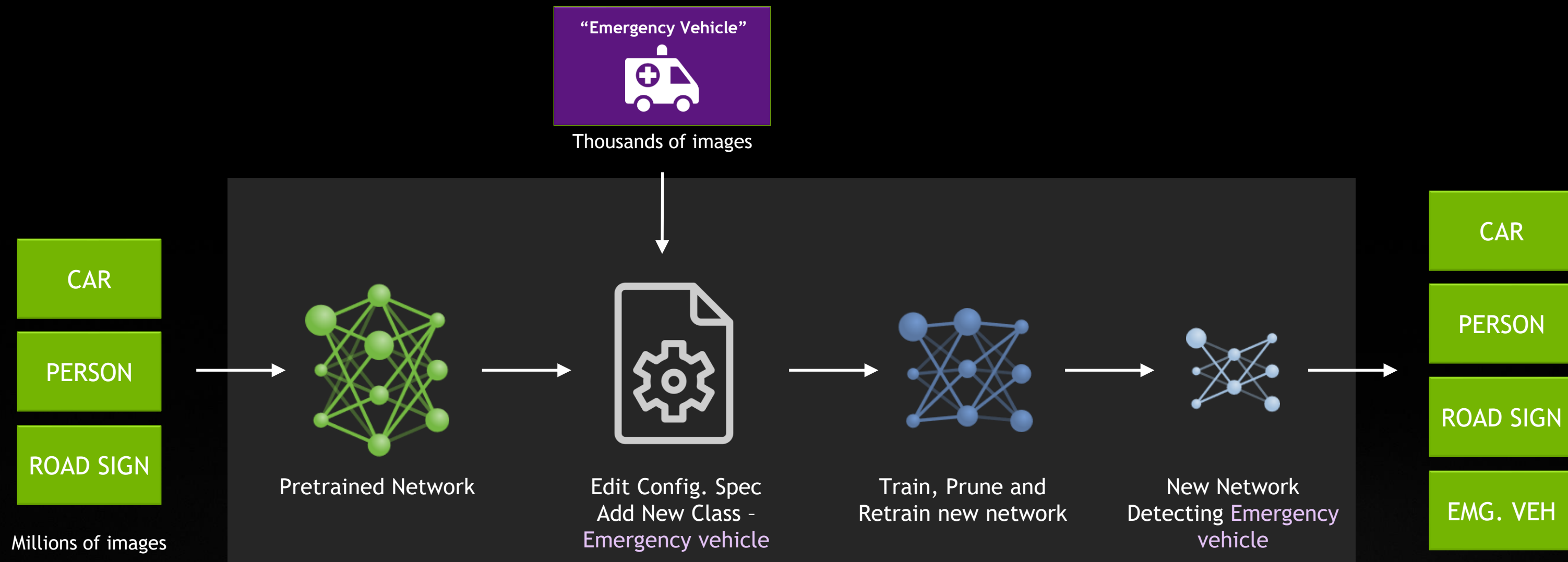
- 1 Reduce model size
- 2 Incrementally retrain model after pruning to recover accuracy



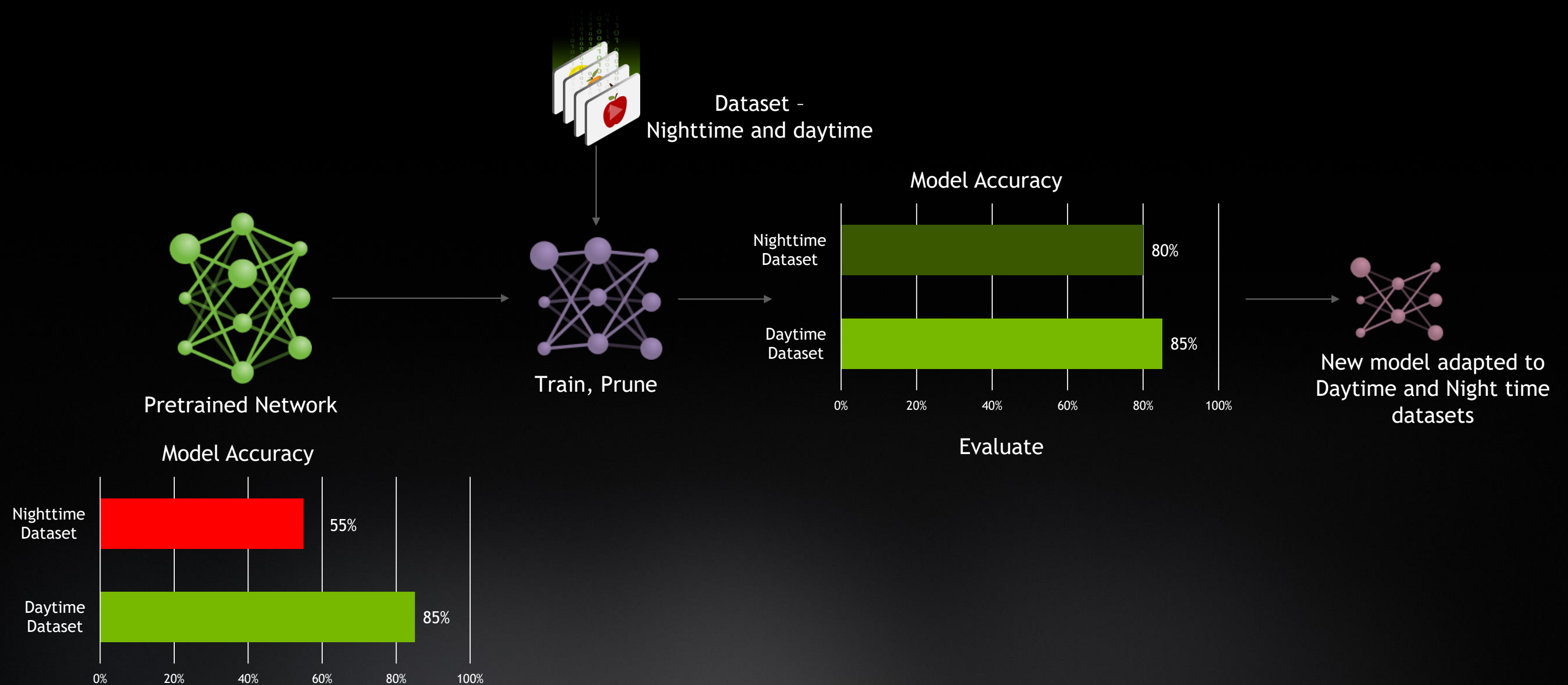
Network - TrafficCamNet



ADDING NEW CLASSES



SCENE ADAPTATION



ZERO-CODING TRAINING

Train with pre-built Jupyter Notebooks

Object Detection using TLT DetectNet_v2

Transfer learning is the process of transferring learned features from one application to another. It is a commonly used training technique where you use a model trained on one task and re-train to use it on a different task.

Transfer Learning Toolkit (TLT) is a simple and easy-to-use Python based AI toolkit for taking purpose-built AI models and customizing them with users' own data.

- ### Learning Objectives
- In this notebook, you will learn how to leverage the simplicity of:
- Take a pretrained resnet18 model and train a ResNet-18
 - Prune the trained detectnet_v2 model
 - Retrain the pruned model to recover lost accuracy
 - Export the pruned model
 - Quantize the pruned model using QAT
 - Run inference on the trained model
 - Export the pruned, quantized and retrained model to a .etf
 - Run inference on the exported .etf model to verify deployment

0. Set up env variables and map drives

When using the purpose-built pretrained models from NGC, please make sure to set the \$LOCAL_PROJECT_DIR env variable. Failing to do so, can lead to errors when trying to load them as pretrained models.

The following notebook requires the user to set an env variable called the \$LOCAL_PROJECT_DIR. The dataset to run this notebook is expected to reside in the \$LOCAL_PROJECT_DIR/data, while the \$LOCAL_PROJECT_DIR/detectnet_v2. More information on how to set up the dataset and subsequent cells.

Note: Please make sure to remove any stray artifacts/files from the \$USER_EXPERIMENT_DIR. Having checkpoint files etc may interfere with subsequent training runs.

Note: This notebook currently is by default set up to run training using 1 GPU. To use more GPUs, please refer to the README file.

```
In [ ]: # Setting up env variables for cleaner command line commands.
import os

%env KEY=tl1_encode
%env NUM_GPUS=1
%env USER_EXPERIMENT_DIR=/workspace/tlt-experiments/detectnet_v2
%env DATA_DOWNLOAD_DIR=/workspace/tlt-experiments/data

# Set this path if you don't run the notebook from the samples directory.
%env NOTEBOOK_ROOT=/tlt-samples/detectnet_v2

# Please define this local project directory that needs to be mapped to the TLT docker session.
# The dataset expected to be present in $LOCAL_PROJECT_DIR/data, while the results for the steps
# in this notebook will be stored at $LOCAL_PROJECT_DIR/detectnet_v2
```

1. Installing the TLT launcher

The TLT launcher is a python package distributed as a python wheel. You may download the latest launcher wheel from [here](#), place it in the same directory as this notebook and install it using the `python3-pip` tool with the by executing the cell mentioned below.

Please note that TLT recommends users to run the TLT launcher in a virtual env with python >=3.6.9. You may follow the instruction in this [page](#) to set up a python virtual env using the `virtualenv` and `virtualenvwrapper` packages.

```
In [ ]: # Uncomment this cell to install the tlt launcher wheel if not previously installed.
# Please make sure to download the latest wheel version from the link mentioned above.
# pip install tlt-0.0.4.dev0-py3-none-any.whl

In [ ]: # Initialize the tlt launcher
tlt init
```

2. Prepare dataset and pre-trained model

We will be using the kitti object detection dataset for this example. To find the dataset, please refer to the [kitti website](#). Please download both, the left color images of the dataset and the right grayscale images. Please place the zip files in \$LOCAL_DATA_DIR.

The data will then be extracted to have

- training images in \$LOCAL_DATA_DIR/training/image_2
- training labels in \$LOCAL_DATA_DIR/training/label_2
- testing images in \$LOCAL_DATA_DIR/testing/image_2

You may use this notebook with your own dataset as well. To use this notebook with your own dataset, please refer to the README file.

Note: There are no labels for the testing images, therefore we use it just for inference.

3. Provide training specification

- Tfrecords for the train datasets
 - To use the newly generated tfrecords, update the dataset_config parameter in the spec file at \$SPEC_DIR/detectnet_v2_train_resnet18_kitti.txt
 - Update the fold number to use for evaluation. In case of random data split, please use fold 0 only
 - For sequence-wise split, you may use any fold generated from the dataset convert tool
- Pre-trained models
- Augmentation parameters for on the fly data augmentation
- Other training (hyper-)parameters such as batch size, number of epochs, learning rate etc.

```
In [ ]: # cat $LOCAL_SPEC_DIR/detectnet_v2_train_resnet18_kitti.txt
```

4. Run TLT training

- Provide the sample spec file and the output directory location for models

Note: The training may take hours to complete. Also, the remaining notebook, assumes that the training was done in single-GPU mode. When run in multi-GPU mode, please expect to update the pruning and inference steps with new pruning thresholds and updated parameters in the clusterfile.json accordingly for optimum performance.

Detectnet_v2 now supports restart from checkpoint. In case the training job is killed prematurely, you may resume training from the closest checkpoint by simply re-running the same command line. Please do make sure to use the **same number of GPUs** when restarting the training.

When running the training with NUM_GPUS>1, you may need to modify the `batch_size_per_gpu` and `Learning_rate` to get similar mAP as a 1GPU training run. In most cases, scaling down the batch-size by a factor of NUM_GPUS or scaling up the learning rate by a factor of NUM_GPUS would be a good place to start.

TRANSFER LEARNING TOOLKIT 2.0

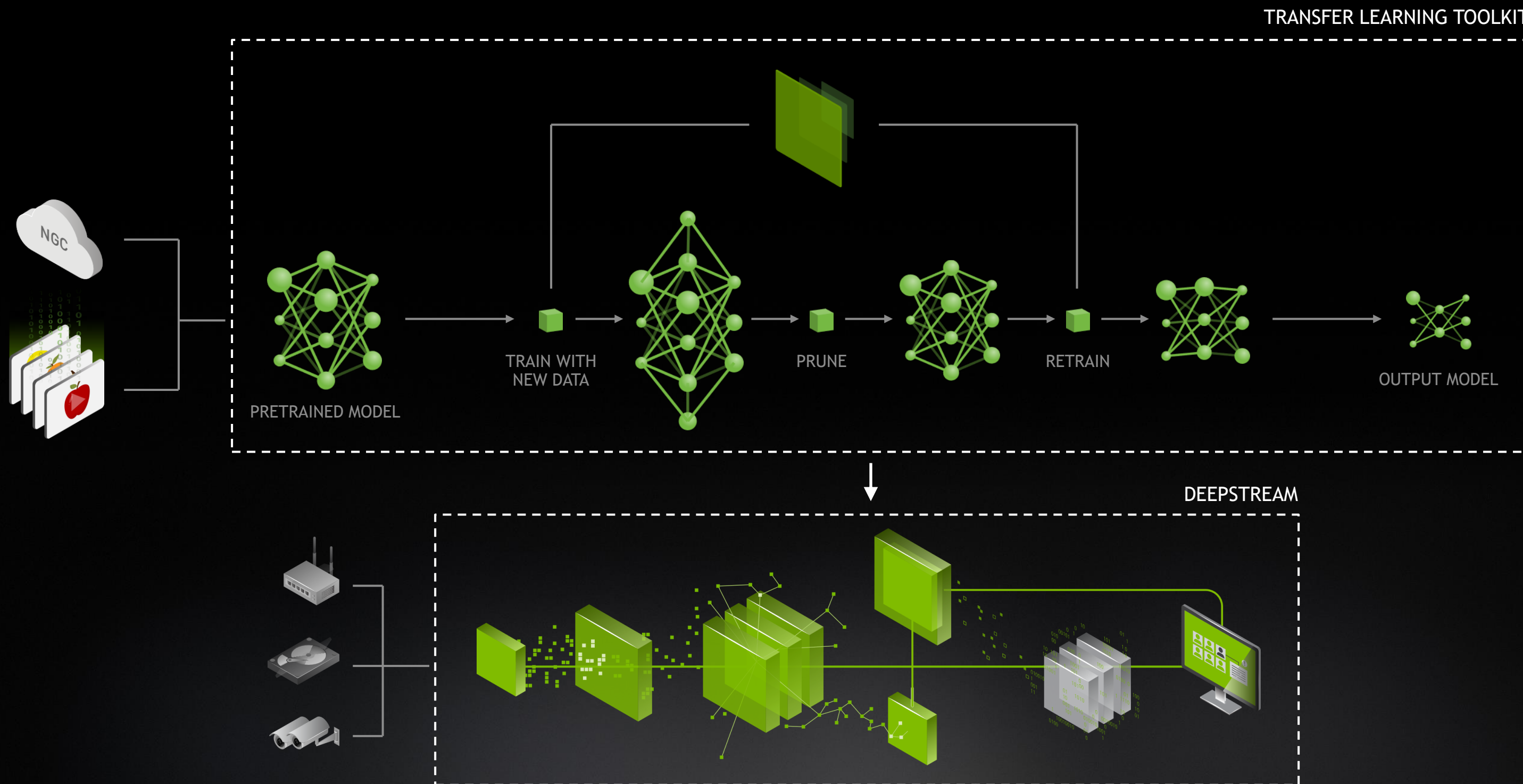
	Classification	Object Detection					
Network Architecture		DetectNet_V2	FasterRCNN	SSD	YOLOV3 [†]	RetinaNet [†]	DSSD [†]
Supported Backbones	ResNet 10/18/50	ResNet 10/18/50	ResNet 10/18/50	ResNet 10/18	ResNet 10/18/50	ResNet 10/18/50	ResNet 10/18/50
	VGG16/19	VGG16/19	VGG16/19	VGG16/19 [†]	VGG16/19	VGG16/19	VGG16/19
	GoogLeNet	GoogLeNet	GoogLeNet	GoogLeNet [†]	GoogLeNet	GoogLeNet	GoogLeNet
	MobileNet V1/V2	MobileNet V1/V2	MobileNet V1/V2	MobileNet V1/V2 [†]	MobileNet V1/V2	MobileNet V1/V2	MobileNet V1/V2
	SqueezeNet			SqueezeNet [†]	SqueezeNet	SqueezeNet	SqueezeNet
	ResNet 34/101 [†]	ResNet 34/101 [†]	ResNet 34/101 [†]	ResNet 50/34/101 [†]	ResNet 34/101	ResNet 34/101	ResNet 34/101
	DarkNet 19/53 [†]		DarkNet 19/53 [†]	DarkNet 19/53 [†]	DarkNet 19/53	DarkNet 19/53	DarkNet 19/53

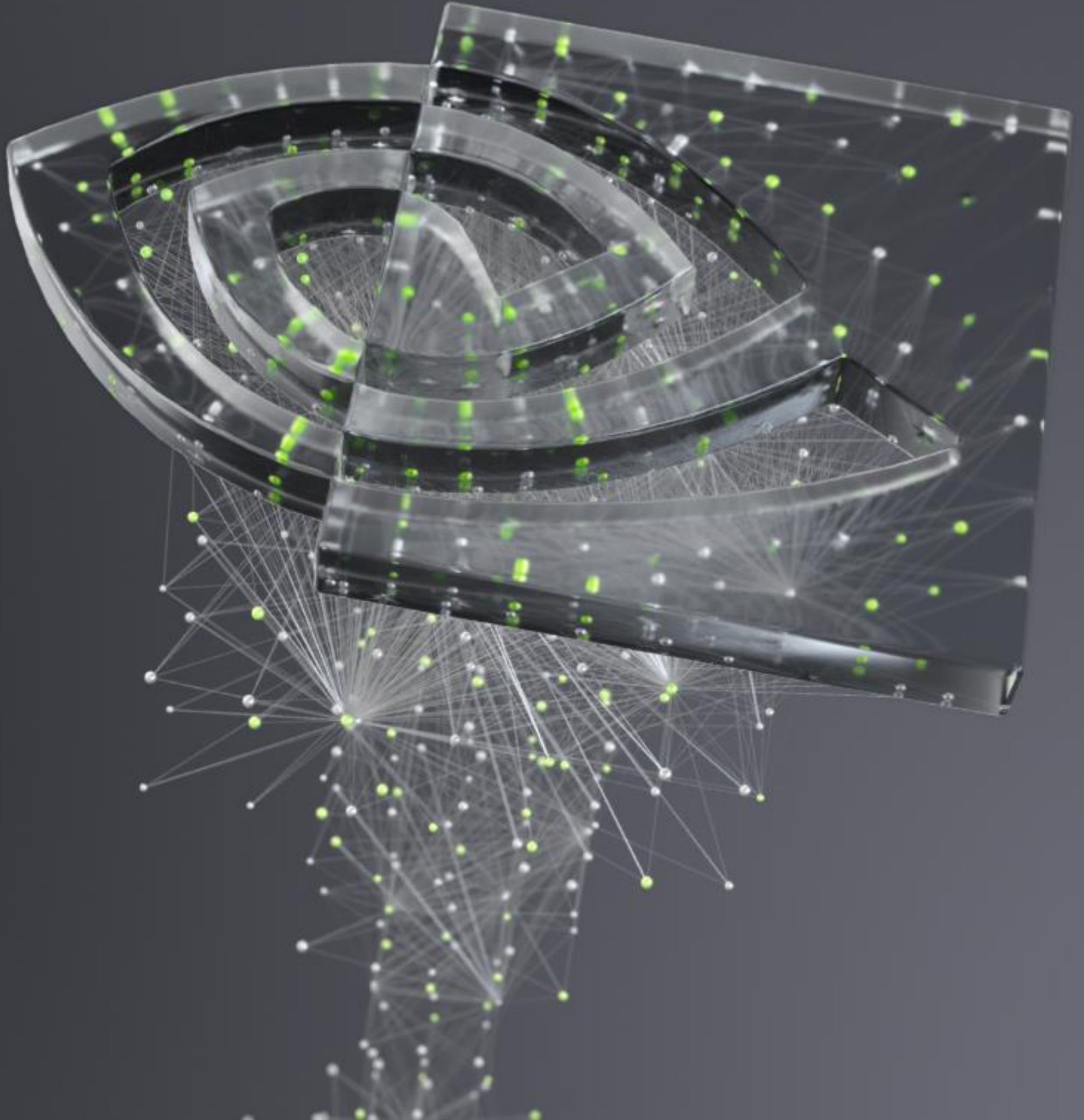
[†] - Available in future release

Models trained on google open images public dataset
Available to download on ngc.nvidia.com

END-TO-END DEEP LEARNING WORKFLOW

Accelerate Time to Market and Save on Compute Resources!





nVIDIA®

ISAAC

March 2020

NVIDIA ISAAC

Isaac Engine

Isaac GEMS

Reference Designs

Isaac Sim

ISAAC SIM

Domain Randomization	Scenario Management	Sensor Models	Robot Models
----------------------	---------------------	---------------	--------------

ISAAC Apps

CARTER (Indoor Robot)	Kaya (Getting Started)	Leonardo (Manipulation)	Custom Applications
-----------------------	------------------------	-------------------------	---------------------

ISAAC GEMS

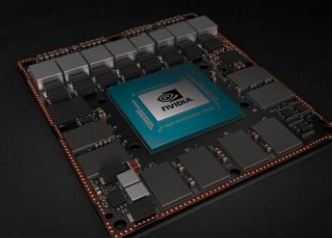
Multi-class Segmentation DNN	3D Object Pose Estimation	Object Detection DNN	Stereo Depth
Stereo Visual Inertial Odometry	Superpixels	AprilTags	2D Skeleton Pose Estimation DNN
DeepStream for Robotics	ORB Feature Tracker	Sensor Drivers	Planning and Control
Image Warping	Navigation	Text to Speech and Keyword Detection DNNs	... and more

ISAAC Engine

Computational Graph & CUDA Messaging	Visualization Tools	Advanced Build System & C API
--------------------------------------	---------------------	-------------------------------

NVIDIA HW Acceleration

TensorRT	cuDNN	CUDA-X	RTX	PhysX
----------	-------	--------	-----	-------



NVIDIA AGX

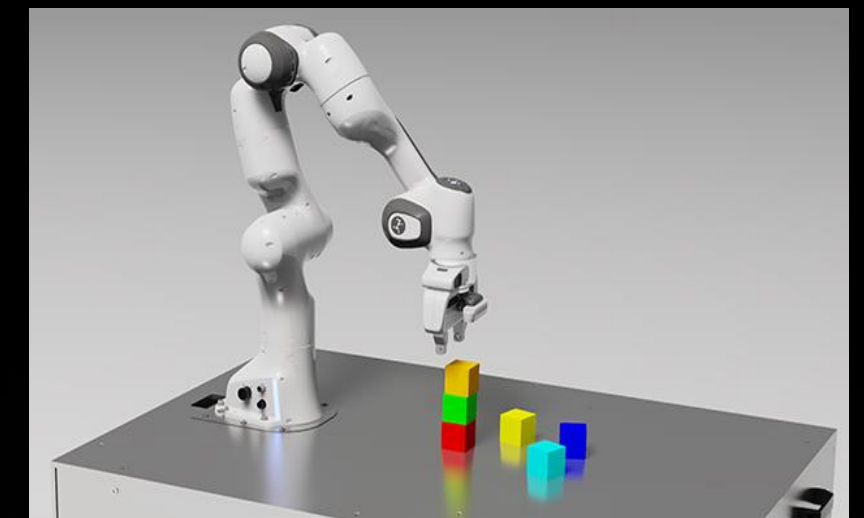


ISAAC SIM

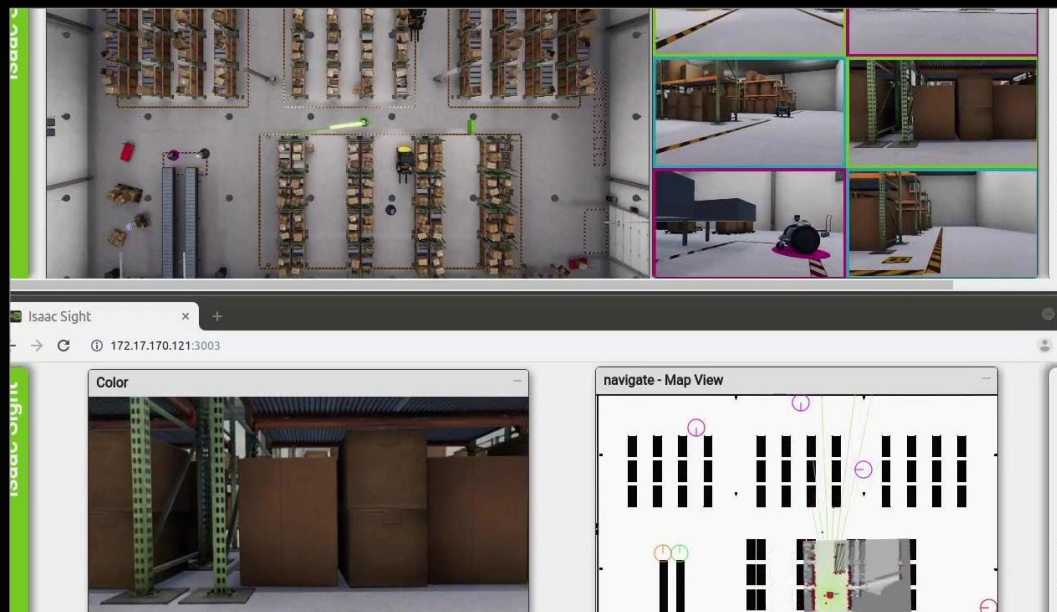
Simulation for Robotics

- ▶ Domain Randomization
- ▶ Scenario Management
- ▶ Import other robots

“Leonardo” Preview Release

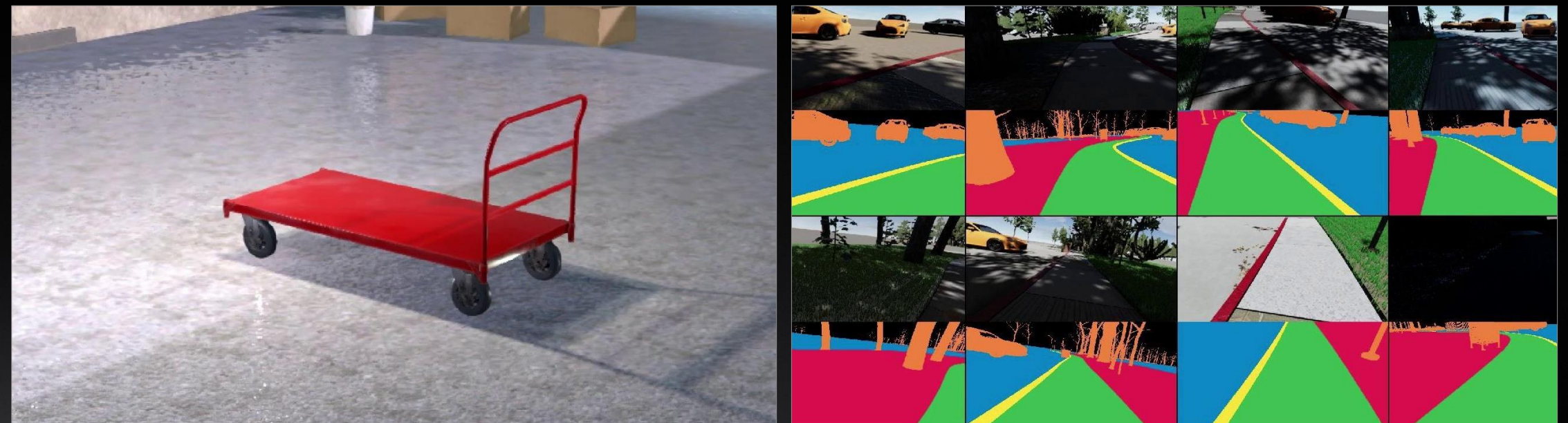


Multi Robot HIL Simulation



Multiple Carter robots operating simultaneously in virtual warehouse; Each operated by an independent Jetson Xavier

ML Training in Simulation



Simulated samples of a dolly (with actual CAD model) used to train object detection and pose estimation neural networks

Procedurally generated simulated images used for segmentation network training



Robotic kitchen assistant

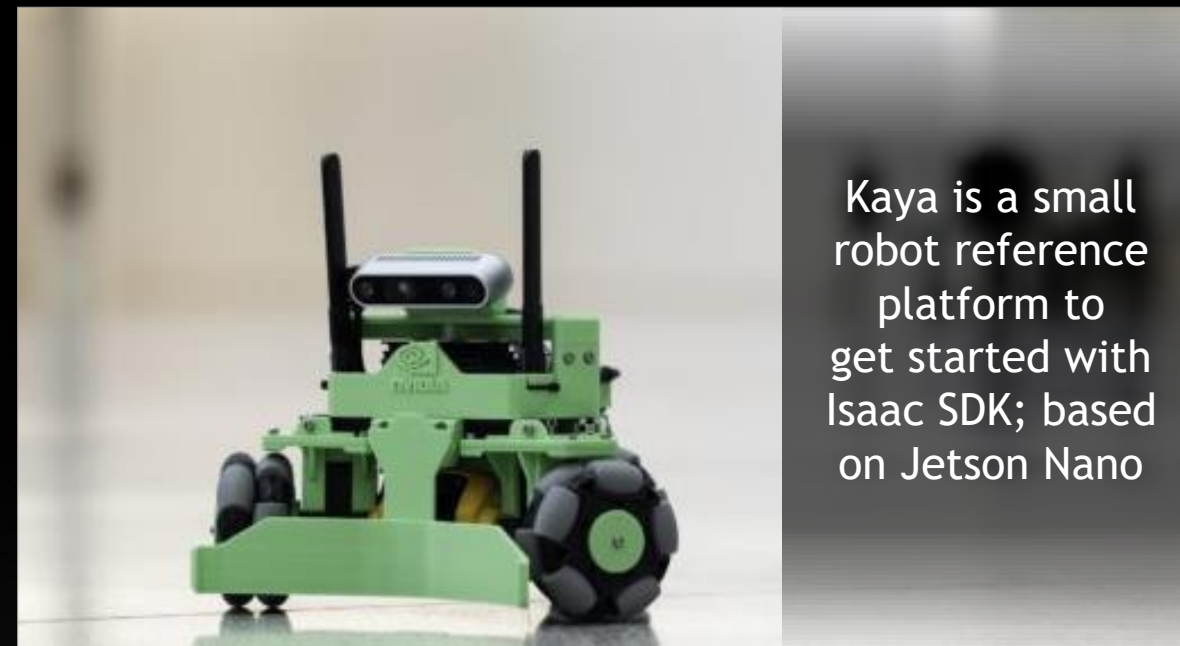
ISAAC REFERENCE DESIGNS

Carter for Indoor Logistics; Kaya to get started; Leonardo for manipulation, more coming...



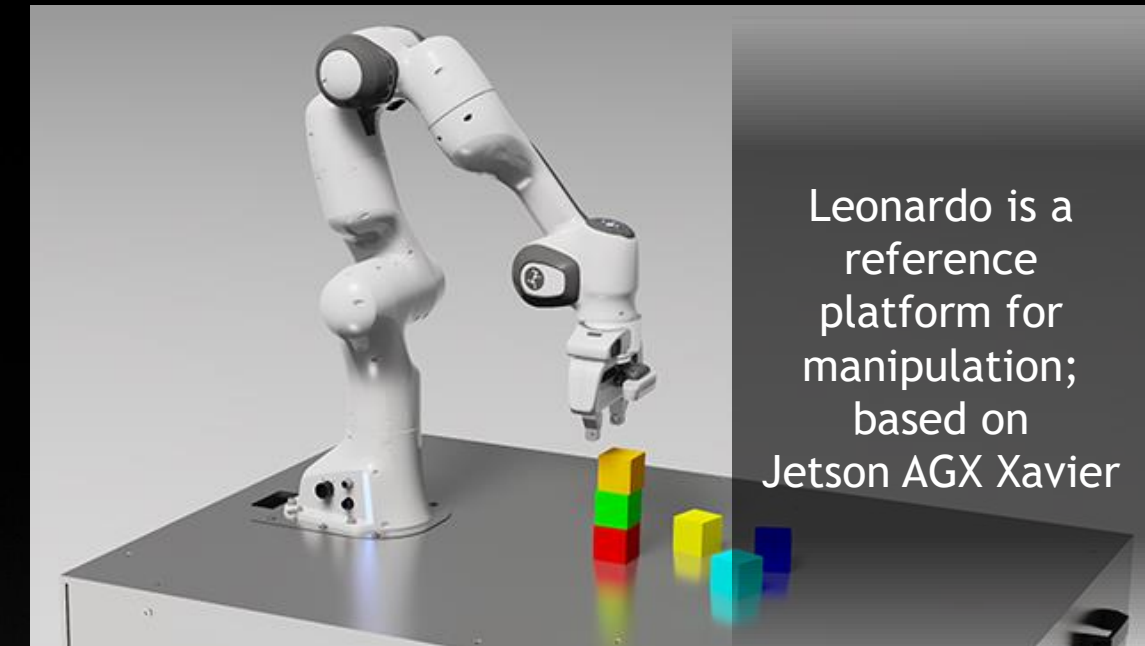
Carter is an Isaac SDK reference robot platform for autonomous indoor delivery and logistics based on the Jetson AGX Xavier platform

CARTER



Kaya is a small robot reference platform to get started with Isaac SDK; based on Jetson Nano

KAYA

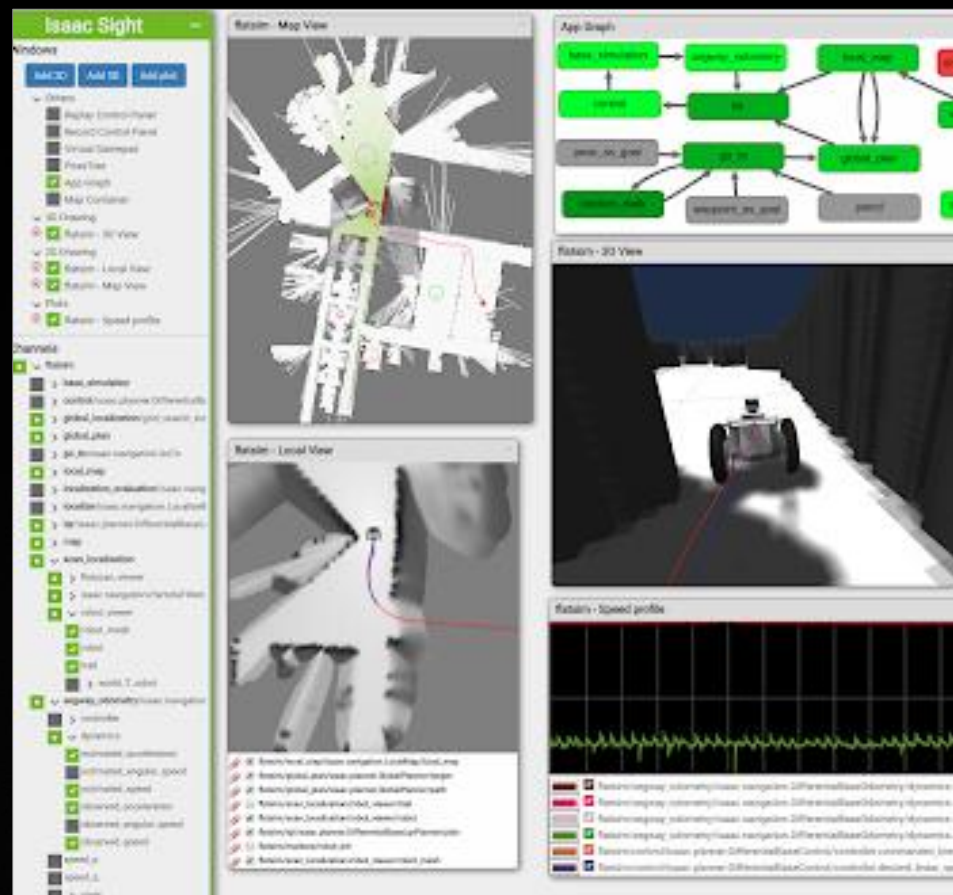


Leonardo is a reference platform for manipulation; based on Jetson AGX Xavier

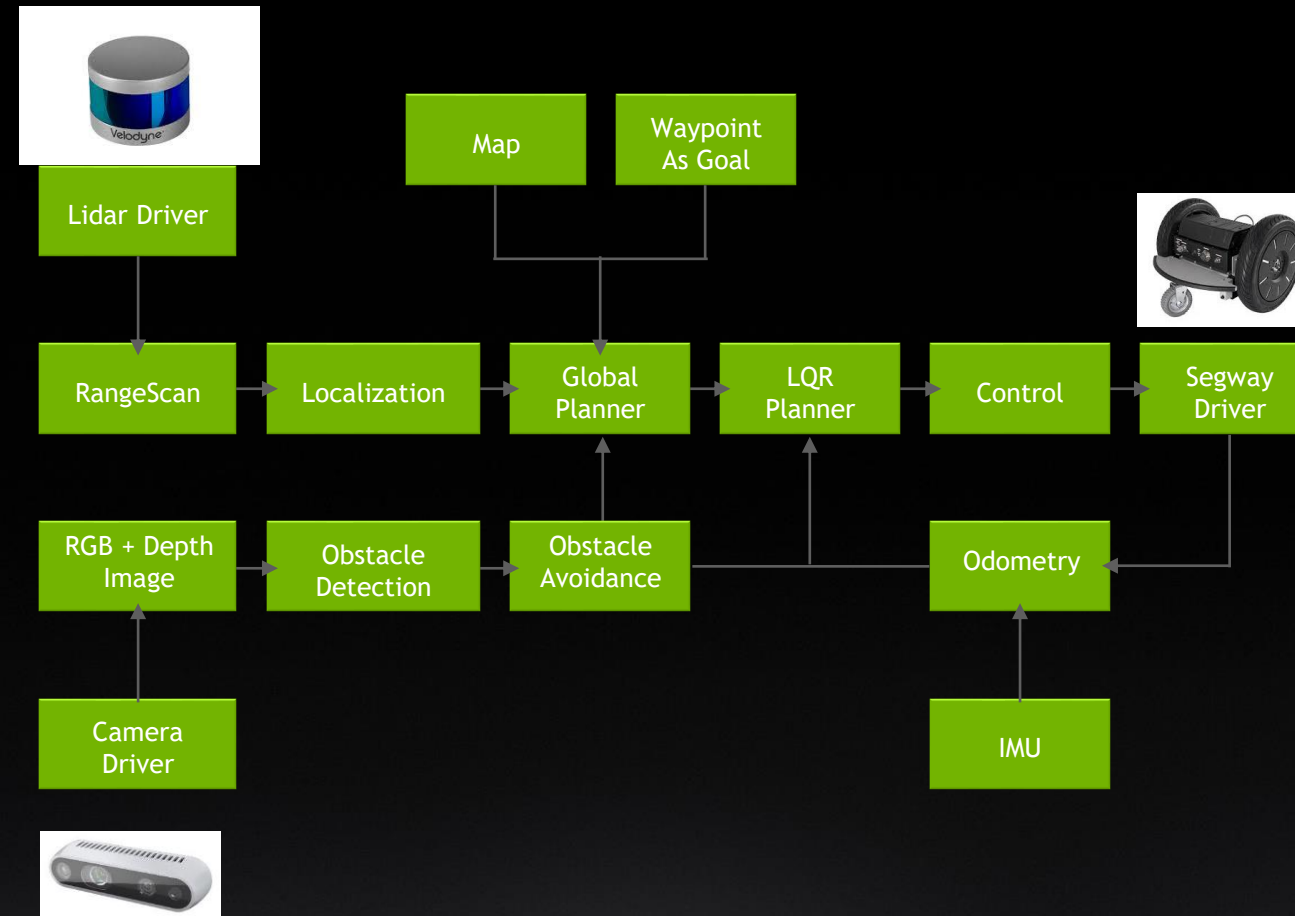
LEONARDO

ISAAC SOFTWARE

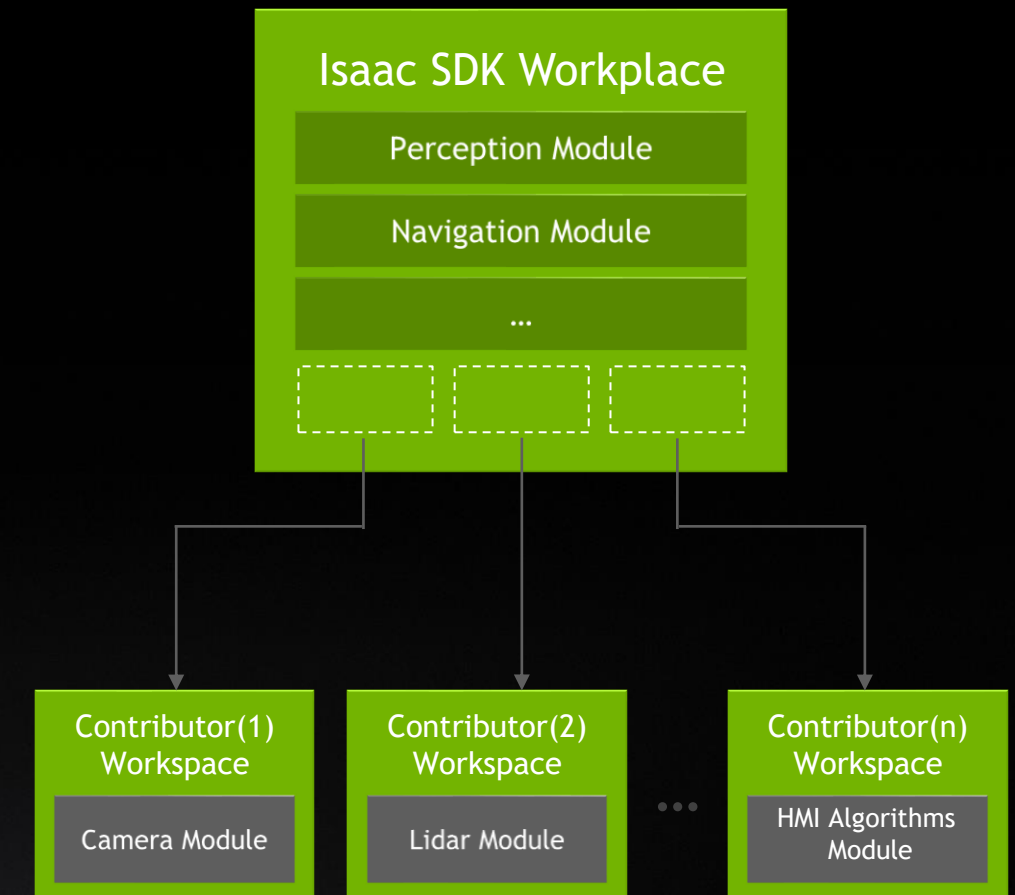
Isaac Engine



Visualization Tool



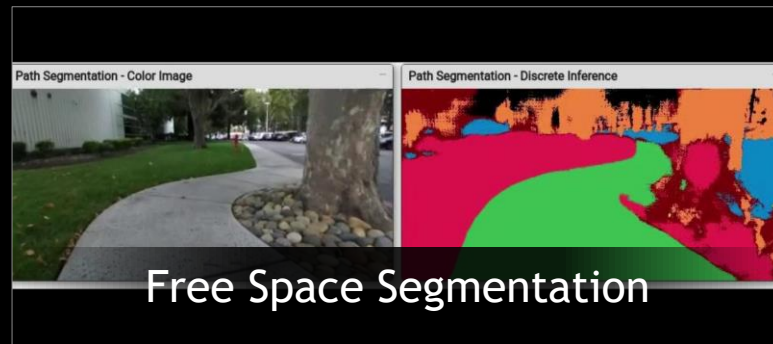
Computational Graph & CUDA Messaging



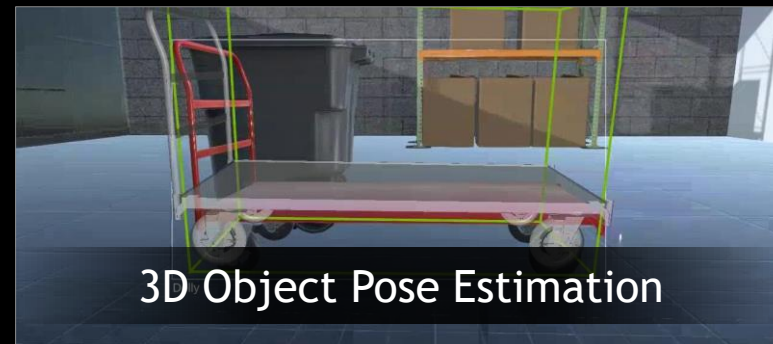
Advanced Build System & C API

ISAAC SOFTWARE

GPU Accelerated Algorithms/DNNs (GEMs)



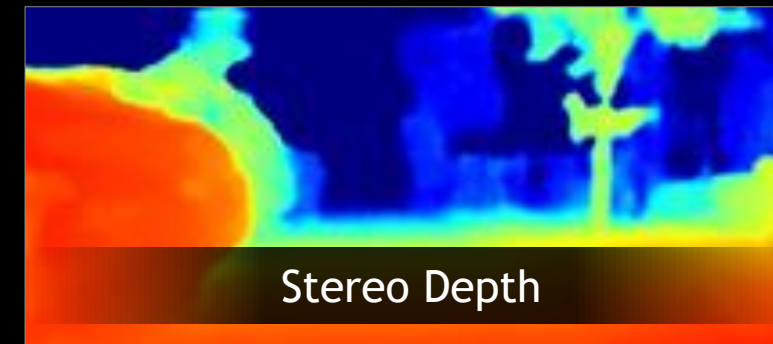
Free Space Segmentation



3D Object Pose Estimation



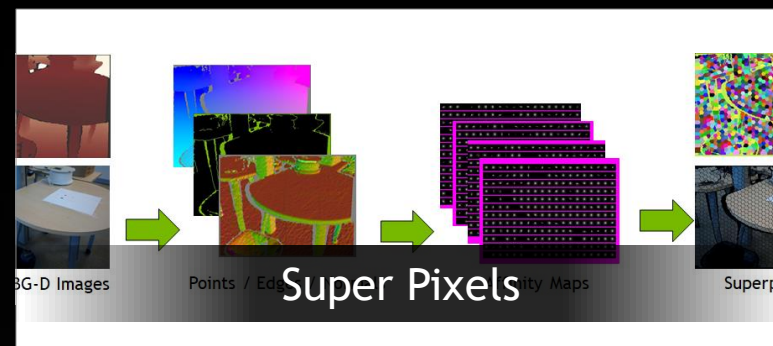
Object Detection



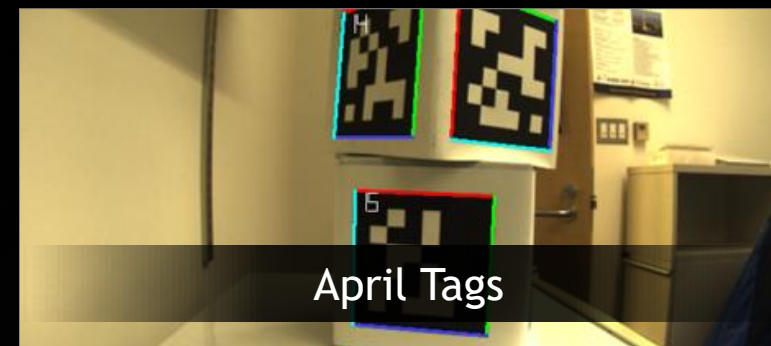
Stereo Depth



Stereo Visual Inertial Odometry



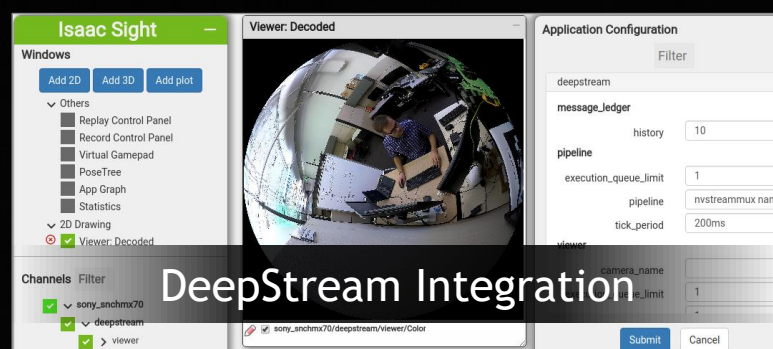
Super Pixels



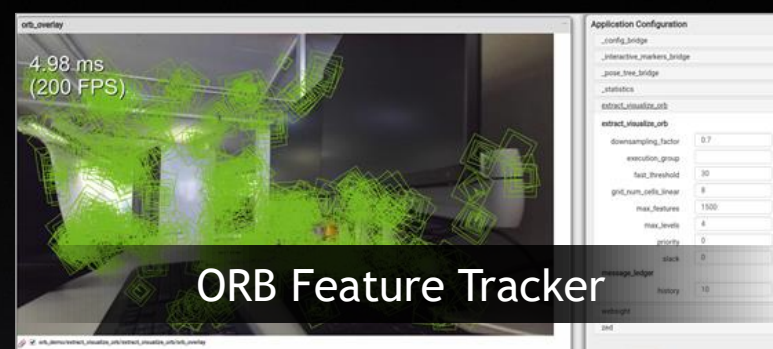
April Tags



2D Skeleton Pose Estimation



DeepStream Integration



ORB Feature Tracker

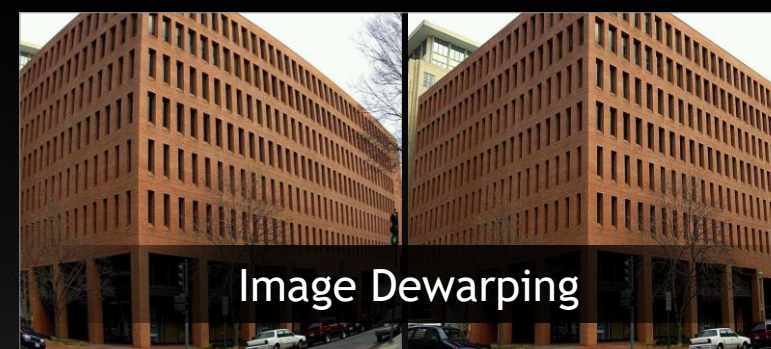
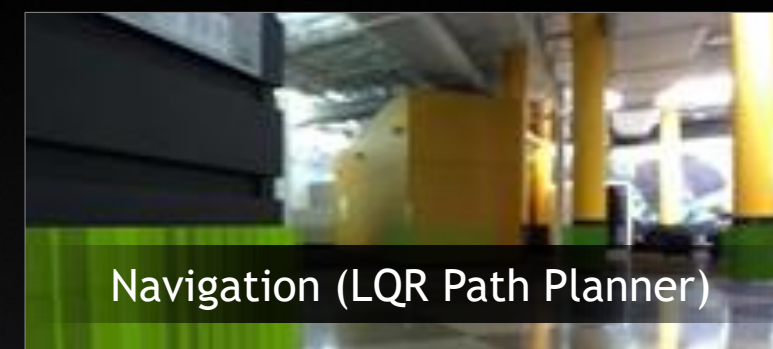


Image Dewarping



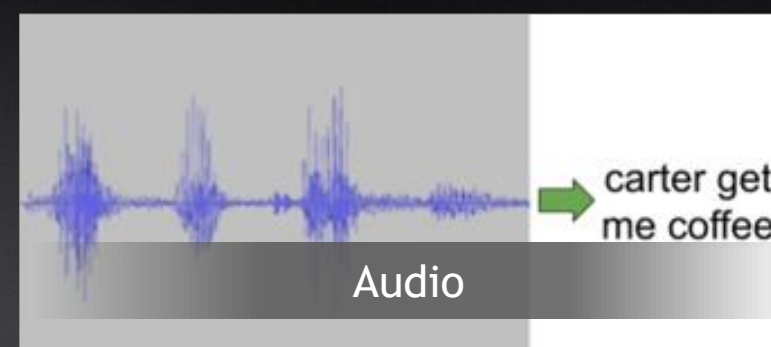
Navigation (LQR Path Planner)



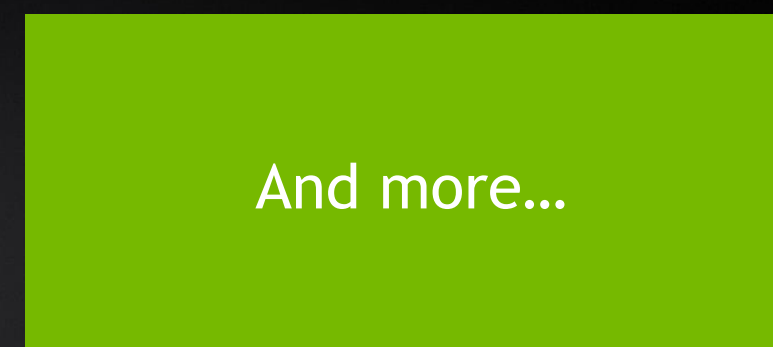
Sensors



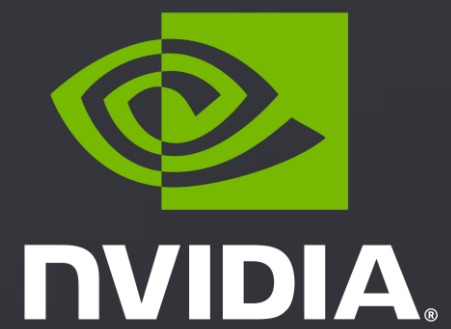
Robot Platforms



Audio

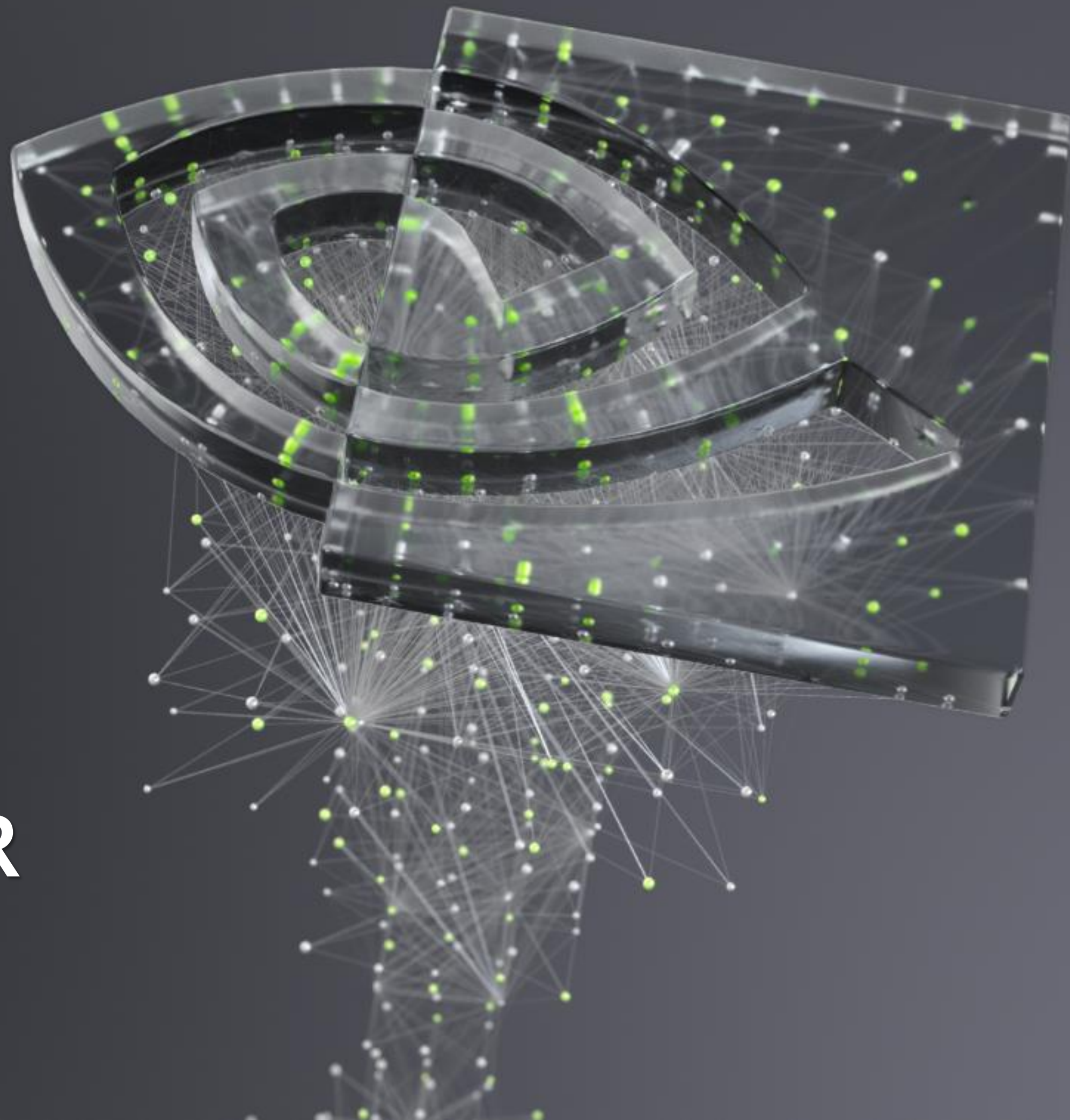


And more...

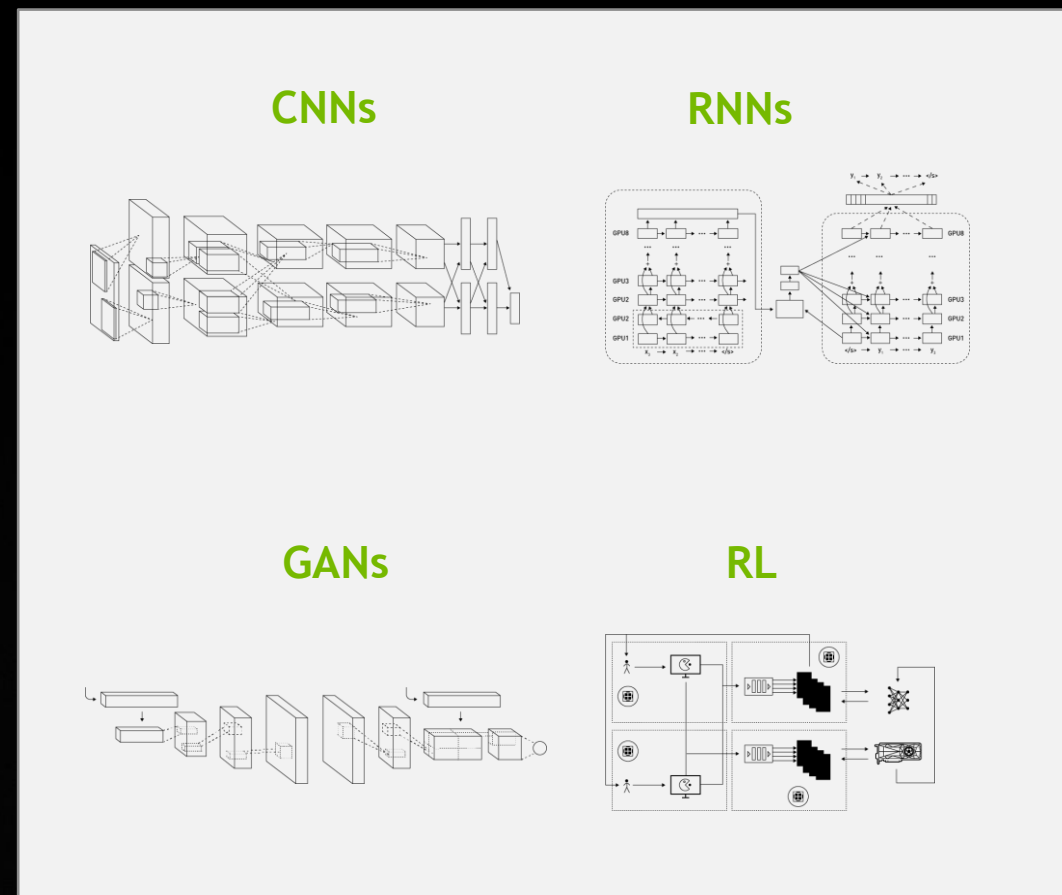


JETSON EGX SERVER

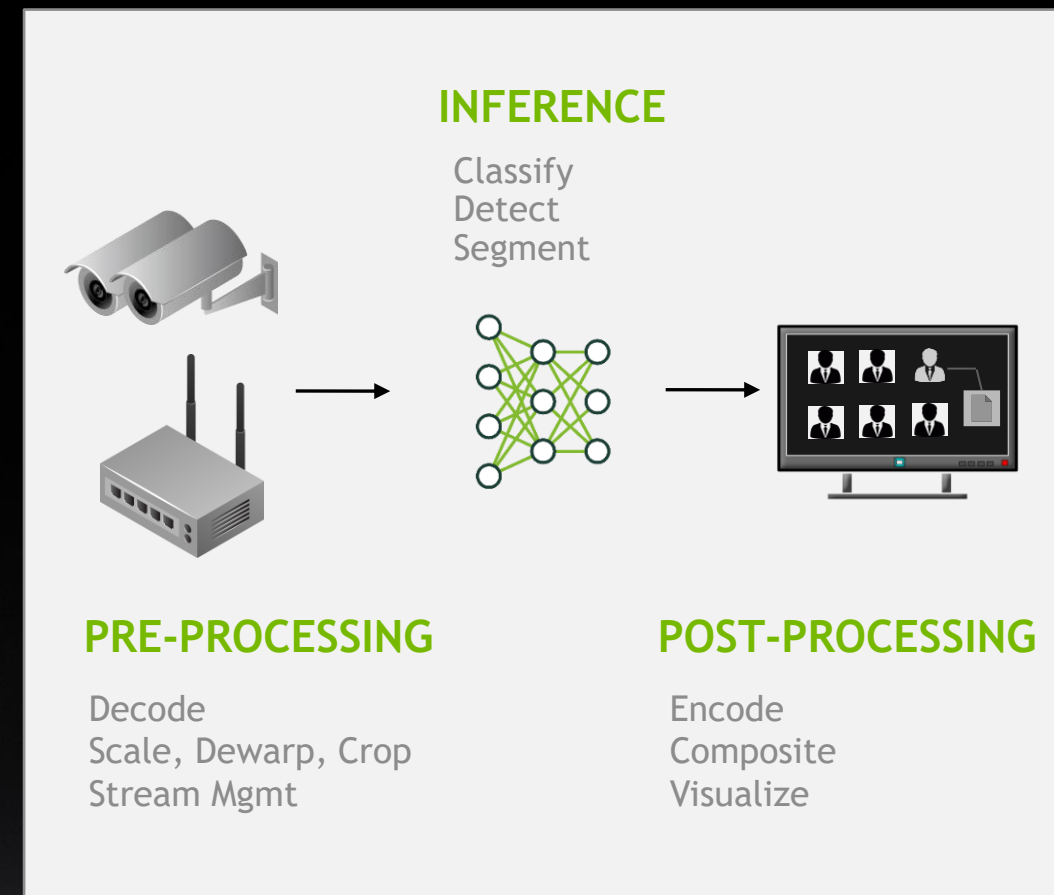
March 2020



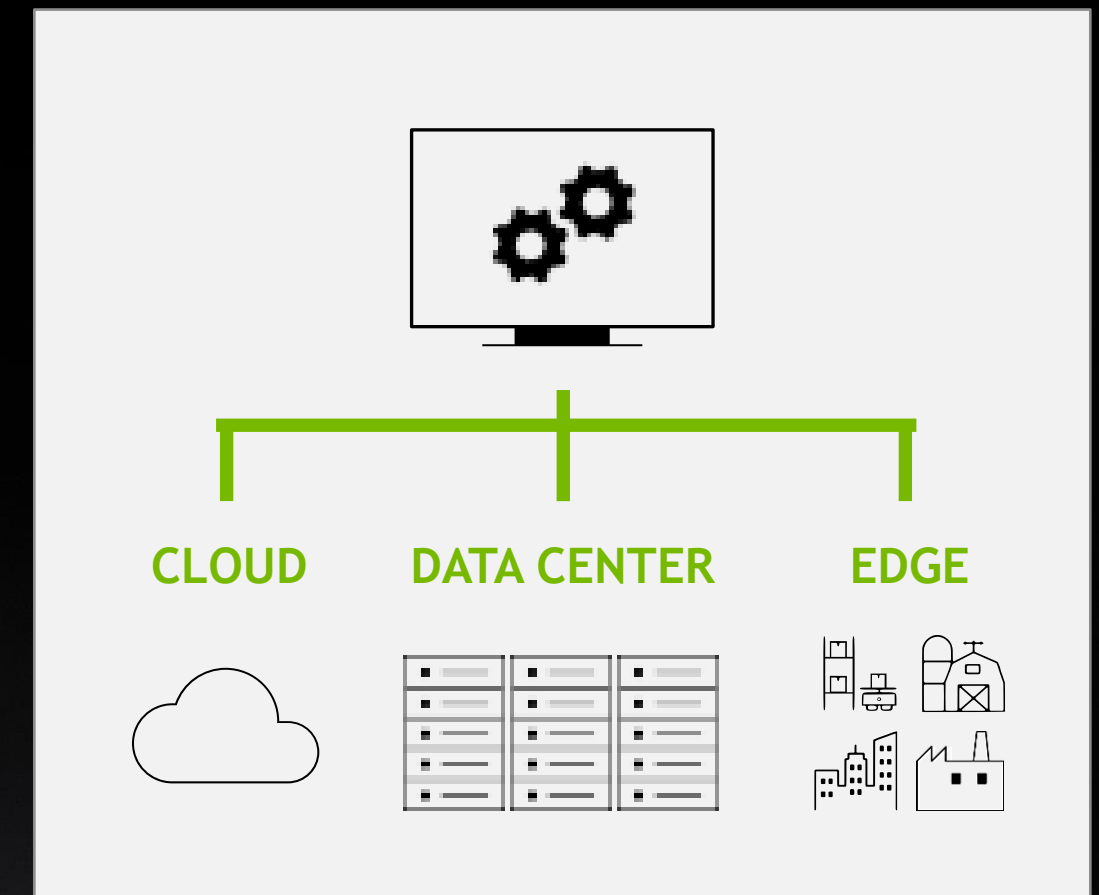
CLOUD NATIVE SUPPORT ON JETSON



AI Inference
Rising Complexity & Diversity

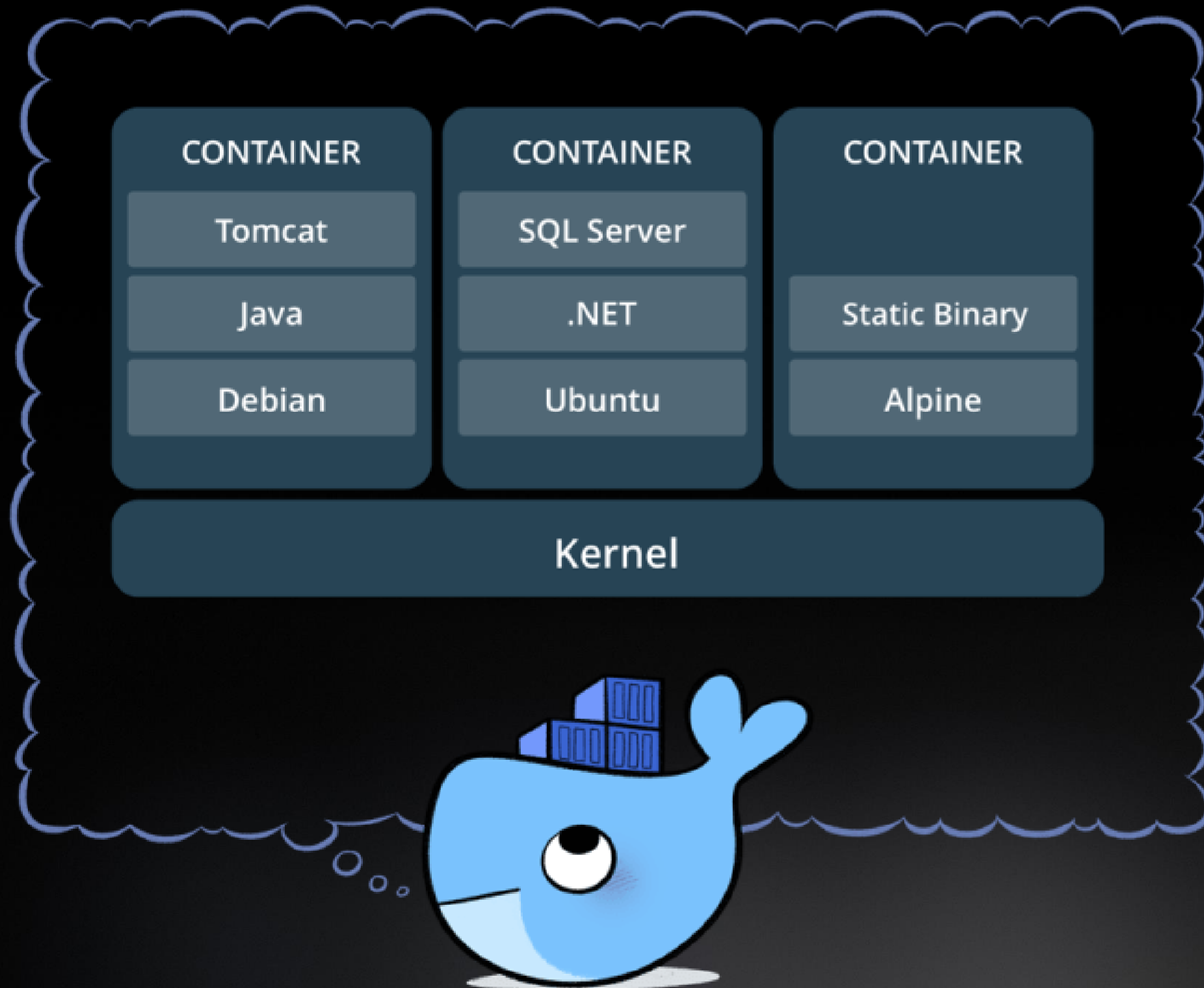


AI Applications
Accelerate the Full Stack



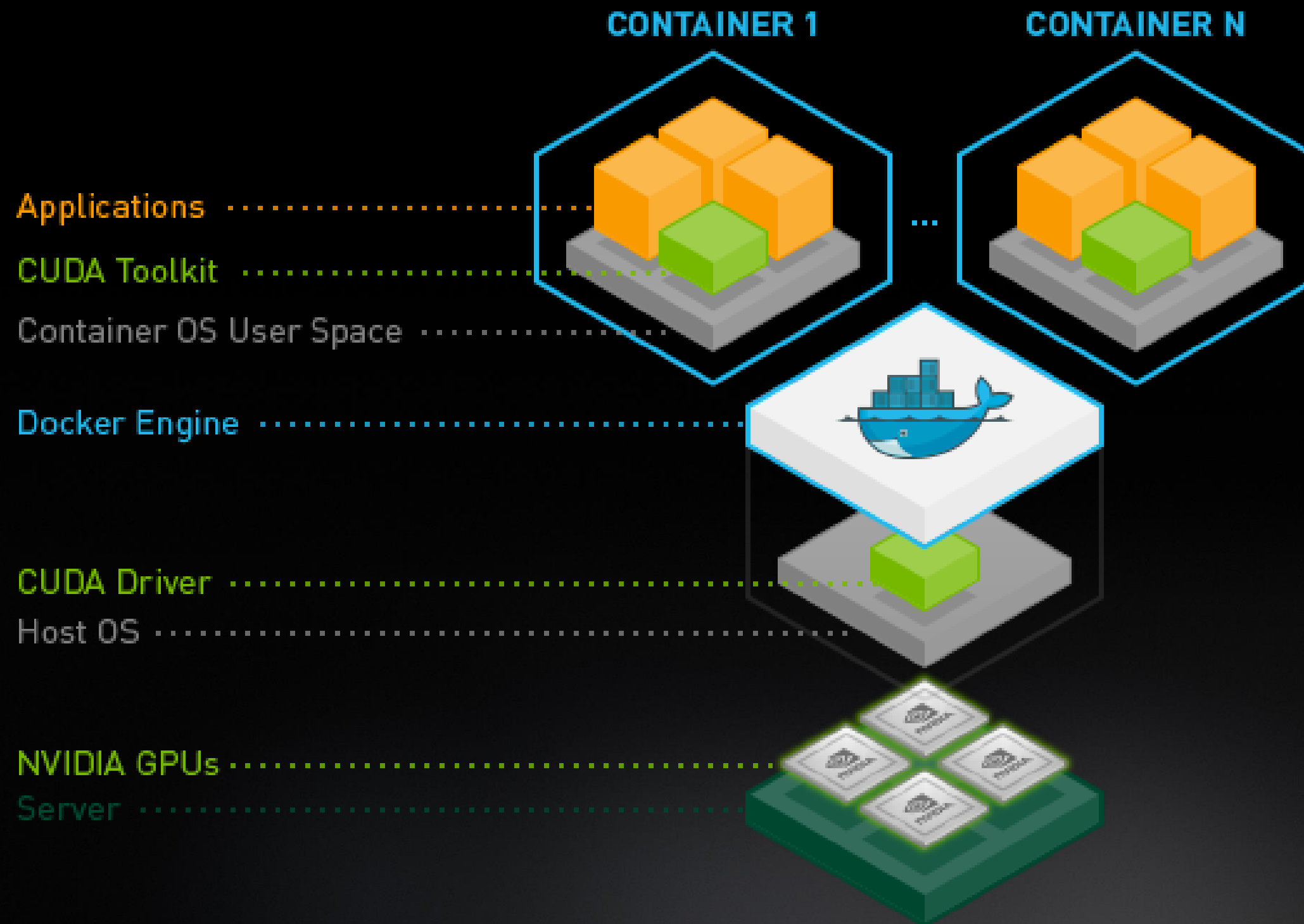
AI Product Lifecycle
Develop, Deploy and Manage at Scale

WHAT IS DOCKER ?



- Docker is a container runtime
- Each container can run a different function
- Each container is isolated from the others
- Containers can be started and stopped, updated, restarted independently

WHY DOCKER ON JETSON ?



- Full support for CUDA acceleration
- Full support for Camera interface, multimedia acceleration, DLA
- Containers can be managed remotely through an Orchestrator (Kubernetes)
- Allow for easy deployment, maintenance and update

BENEFIT OF ORCHESTRATOR

The screenshot displays the Kubernetes Dashboard interface. The top navigation bar includes the Kubernetes logo, a search bar, and a '+ CREATE' button. The breadcrumb trail shows 'Workloads > Pods'. The left sidebar contains a navigation menu with categories: Nodes, Persistent Volumes, Roles, Storage Classes, Namespace (set to 'kube-system'), Overview, Workloads (with sub-items: Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets), and Discovery and Load Balancing.

Two performance charts are visible:

- CPU usage:** A line graph showing CPU usage in cores over time (11:10 to 11:24). The y-axis ranges from 0 to 0.135. The usage fluctuates between approximately 0.09 and 0.12 cores.
- Memory usage:** A line graph showing memory usage in bytes over time. The y-axis ranges from 0 to 644 Mi. The usage is relatively stable, fluctuating between approximately 429 Mi and 572 Mi.

Below the charts is a 'Pods' table with the following data:

Name	Node	Status	Restarts	Age	CPU (cores)	Memory (bytes)
✓ kubernetes-dashboard-7b9c7b	minikube	Running	0	27 minutes	0	19.746 Mi
✓ heapster-qhq6r	minikube	Running	0	27 minutes	0	18.004 Mi
✓ influxdb-grafana-77c7p	minikube	Running	0	27 minutes	0	43.926 Mi
✓ kube-scheduler-minikube	minikube	Running	0	20 hours	0.01	11.930 Mi
✓ etcd-minikube	minikube	Running	0	20 hours	0.015	58.445 Mi



RESOURCES



JETSON DEVELOPER KIT
 Start developing now
 Starting at \$299
[developer.nvidia.com/
 buy-jetson](https://developer.nvidia.com/buy-jetson)



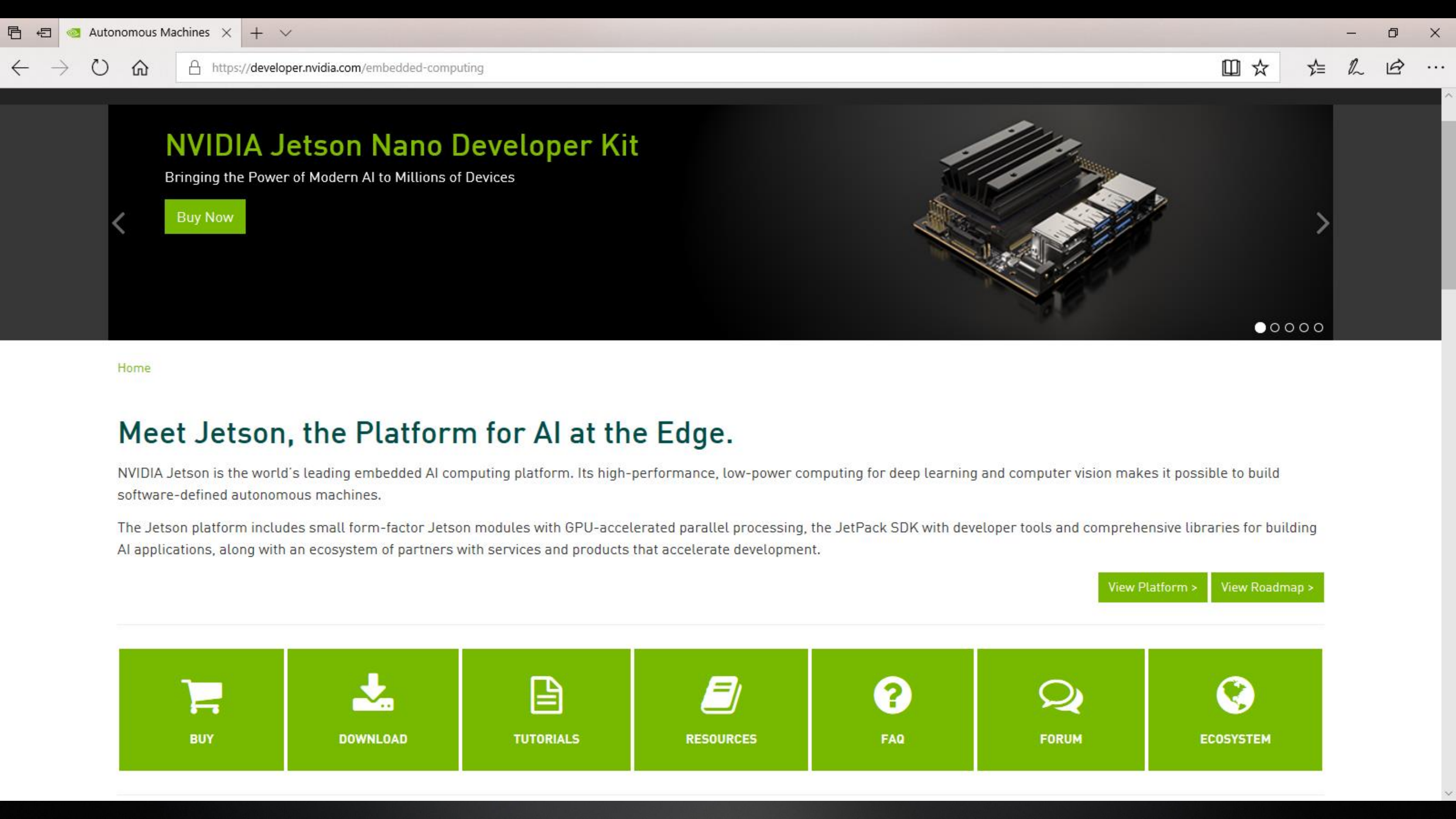
TWO DAYS TO A DEMO
 Create your first demo today
[developer.nvidia.com/
 embedded/twodaystoademo](https://developer.nvidia.com/embedded/twodaystoademo)



DEEP LEARNING INSTITUTE
 Training • Labs Nanodegrees
nvidia.com/DLI



GTC
 Largest event for GPU
 developers
[https://www.nvidia.com/
 en-us/gtc/](https://www.nvidia.com/en-us/gtc/)



NVIDIA Jetson Nano Developer Kit

Bringing the Power of Modern AI to Millions of Devices

Buy Now



Home

Meet Jetson, the Platform for AI at the Edge.

NVIDIA Jetson is the world's leading embedded AI computing platform. Its high-performance, low-power computing for deep learning and computer vision makes it possible to build software-defined autonomous machines.

The Jetson platform includes small form-factor Jetson modules with GPU-accelerated parallel processing, the JetPack SDK with developer tools and comprehensive libraries for building AI applications, along with an ecosystem of partners with services and products that accelerate development.

[View Platform >](#)

[View Roadmap >](#)



BUY



DOWNLOAD



TUTORIALS



RESOURCES



FAQ



FORUM



ECOSYSTEM

