HELLO AI WORLD — MEET JETSON NANO

WEBINAR AGENDA

Intro to Jetson Nano

- AI for Autonomous Machines
- Jetson Nano Developer Kit
- Jetson Nano Compute Module

Jetson Software

- JetPack 4.2
- ML/DL Framework Support
- NVIDIA TensorRT
- Inferencing Benchmarks

Application SDKs

- DeepStream SDK
- Isaac Robotics SDK

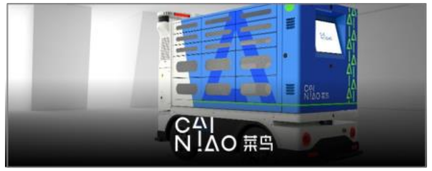
Getting Started

- Jetson Nano Resources
- Hello AI World
- JetBot
- System Setup
- Tips and Tricks

JETSON POWERS AUTONOMOUS MACHINES

WAREHOUSE

DELIVERY



AGRICULTURE

RETAIL

INDUSTRIAL

JETSON NANO DEVELOPER KIT

$99 CUDA-X AI Computer

128 CUDA Cores | 4 Core CPU
4GB LPDDR4 Memory
472 GFLOPs
5W | 10W

Accessible and easy to use

JETSON NANO DEVKIT SPECS

	PROCESSOR
CPU	64-bit Quad-core ARM A57 @ 1.43GHz
GPU	128-core NVIDIA Maxwell @ 921MHz
Memory	4GB 64-bit LPDDR4 @ 1600MHz | 25.6GB/s
Video Encoder	4Kp30 | (4x) 1080p30 | (2x) 1080p60
Video Decoder	4Kp60 | (2x) 4Kp30 | (8x) 1080p30 | (4x) 1080p60

	INTERFACES
USB	(4x) USB 3.0 A (Host) | USB 2.0 Micro B (Device)
Camera	MIPI CSI-2 x2 (15-position Flex Connector)
Display	HDMI | DisplayPort
Networking	Gigabit Ethernet (RJ45, PoE)
Wireless	M.2 Key-E with PCIe x1
Storage	MicroSD card (16GB UHS-1 recommended minimum)
40-Pin Header	UART | SPI | I2C | I2S | Audio Clock | GPIOs
Power	5V DC (μUSB, Barrel Jack, PoE) - 5W | 10W
Size	80x100mm

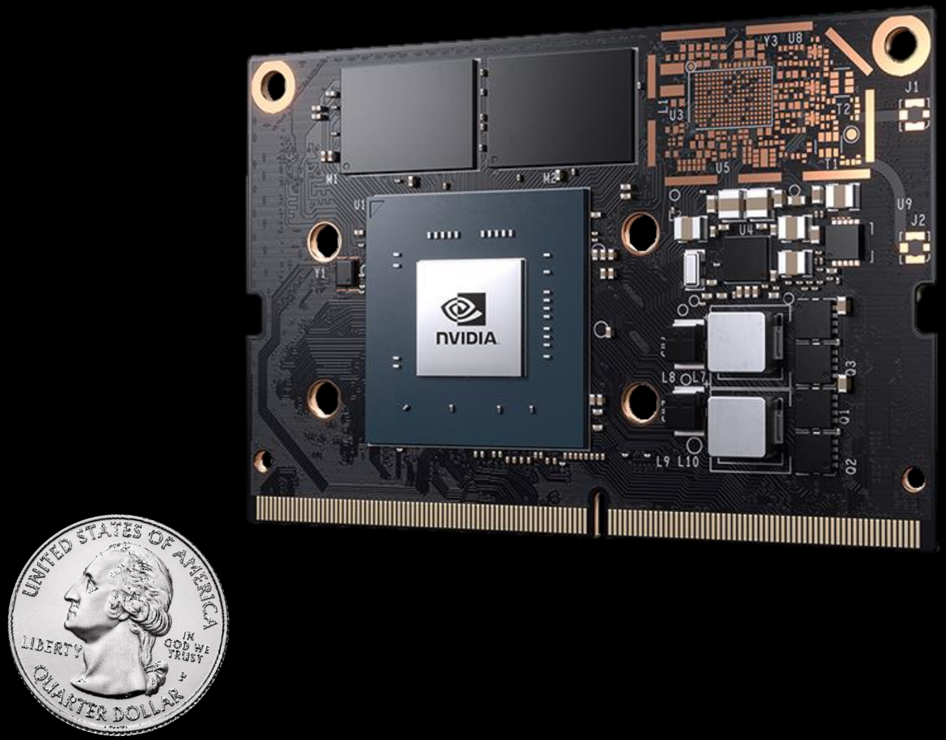
Distributors Include: a ARROW newegg NVIDIA seeed Siliconhighway sparkfun ELECTRONICS

JETSON NANO

Compact AI Compute Module

128 CUDA Cores | 4 Core CPU
4GB LPDDR4 Memory
16GB eMMC 5.1
45x70mm
5W | 10W

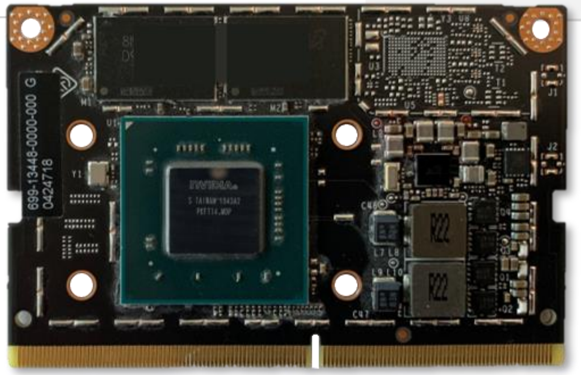
$129 (1Ku)
Available June 2019



JETSON NANO COMPUTE MODULE

PROCESSOR	
CPU	64-bit Quad-core ARM A57 @ 1.43GHz
GPU	128-core NVIDIA Maxwell @ 921MHz
Memory	4GB 64-bit LPDDR4 @ 1600MHz | 25.6GB/s
Video Encoder	4Kp30 | (4x) 1080p30 | (2x) 1080p60
Video Decoder	4Kp60 | (2x) 4Kp30 | (8x) 1080p30 | (4x) 1080p60

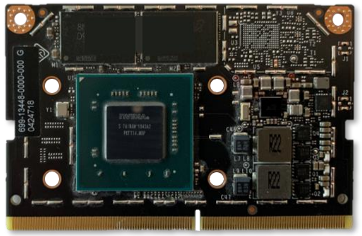
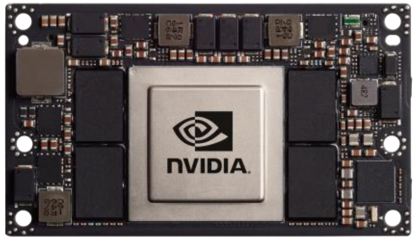
INTERFACES	
USB	USB 3.0 | (3x) USB 2.0
Camera	12 lanes MIPI CSI-2 (up to 4 cameras)
Display	HDMI | DP | eDP | DSI
Networking	Gigabit Ethernet
PCIe	PCIe Gen2 x1/x2/x4
Storage	16GB eMMC 5.1
Other I/O	(4x) I2C | (2x) SPI | (3x) UART | (2x) I2S | GPIO
Power	5V DC, 5W | 10W
Size	45x70mm, 260-pin SODIMM connector



Production module
available June 2019

THE JETSON FAMILY

From AI at the Edge to Autonomous Machines

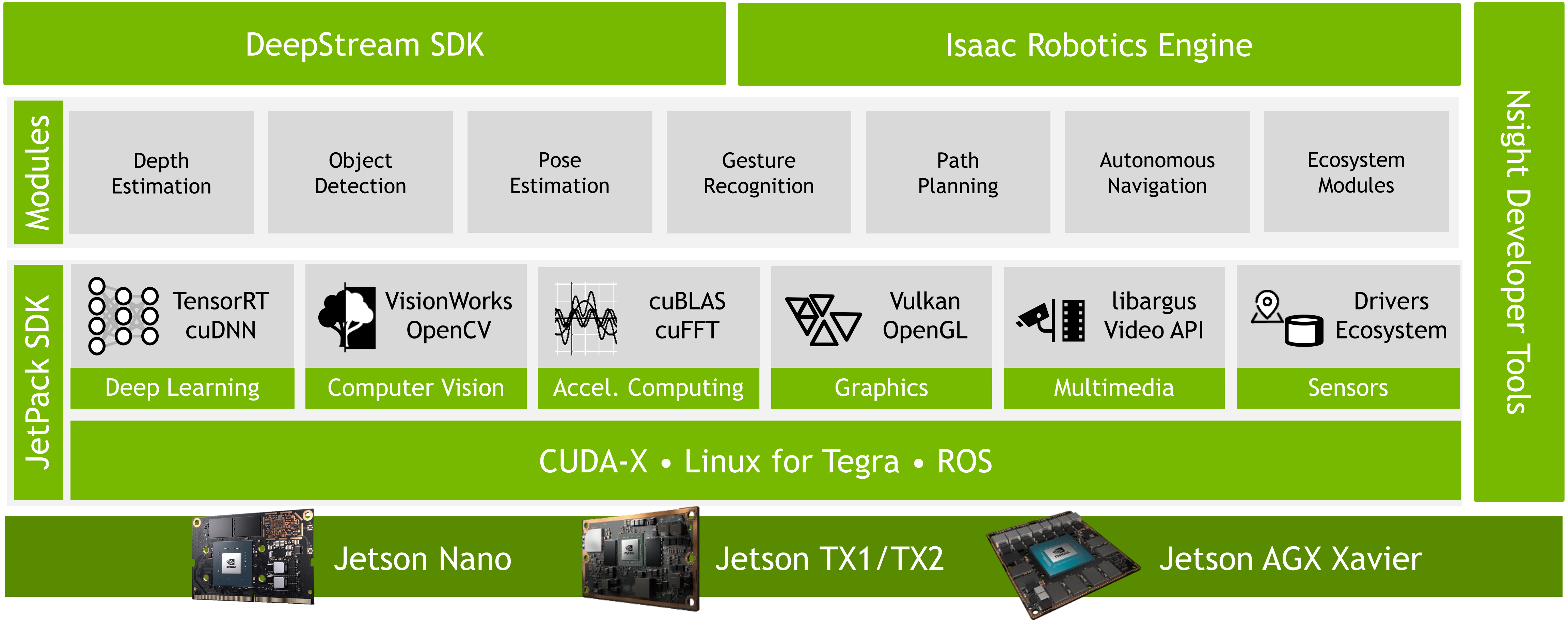
JETSON NANO	JETSON TX1 → JETSON TX2 4GB	JETSON TX2 8GB | Industrial	JETSON AGX XAVIER
5—10W	7—15W	7—15W	10—30W
0.5 TFLOPS (FP16)	1—1.3 TFLOPS (FP16)	1.3 TFLOPS (FP16)	11 TFLOPS (FP16) | 32 TOPS (INT8)
45mm x 70mm	50mm x 87mm	50mm x 87mm	100mm x 87mm
$129 / $99 (Devkit)	$299	$399—$749	$1099

AI at the Edge

Fully Autonomous Machines

Multiple Devices — Same Software

JETSON SOFTWARE



DeepStream SDK

Isaac Robotics Engine

Nsight Developer Tools

Modules

Depth Estimation

Object Detection

Pose Estimation

Gesture Recognition

Path Planning

Autonomous Navigation

Ecosystem Modules

JetPack SDK

TensorRT cuDNN — Deep Learning

VisionWorks OpenCV — Computer Vision

cuBLAS cuFFT — Accel. Computing

Vulkan OpenGL — Graphics

libargus Video API — Multimedia

Drivers Ecosystem — Sensors

CUDA-X • Linux for Tegra • ROS

Jetson Nano

Jetson TX1 / TX2

Jetson AGX Xavier

developer.nvidia.com/jetpack

JETPACK 4.2

Available Now For Jetson

developer.nvidia.com/jetpack

Package Versions

L4T BSP	32.1
Linux Kernel	4.9.140
Vulkan	1.1.1
OpenGL	4.6
OpenGL-ES	3.2.5
EGL	1.5
GLX	1.4
X11 ABI	24
Wayland	1.14
L4T Multimedia API	32.1
Argus Camera API	0.97
GStreamer	1.14.1
Nsight Systems	2019.3
Nsight Graphics	2018.7
Nsight Compute	1.0
Jetson GPIO	1.0
Jetson OS	Ubuntu 18.04
Host OS	Ubuntu 16.04 / 18.04

CUDA	10.0.166
cuDNN	7.3.1.28
TensorRT	5.0.6.3
VisionWorks	1.6
OpenCV	3.3.1
NPP	10.0

Install TensorFlow, PyTorch, Caffe, Caffe2, MXNet, ROS, and other GPU-accelerated libraries

OPEN FRAMEWORK SUPPORT

MACHINE LEARNING

Caffe

Caffe2

K Keras

mxnet

PYTORCH

TensorFlow

ROBOTICS / IOT

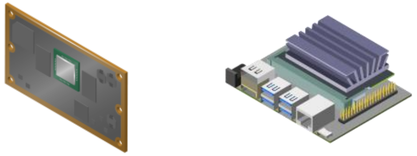
AWS Greengrass

docker

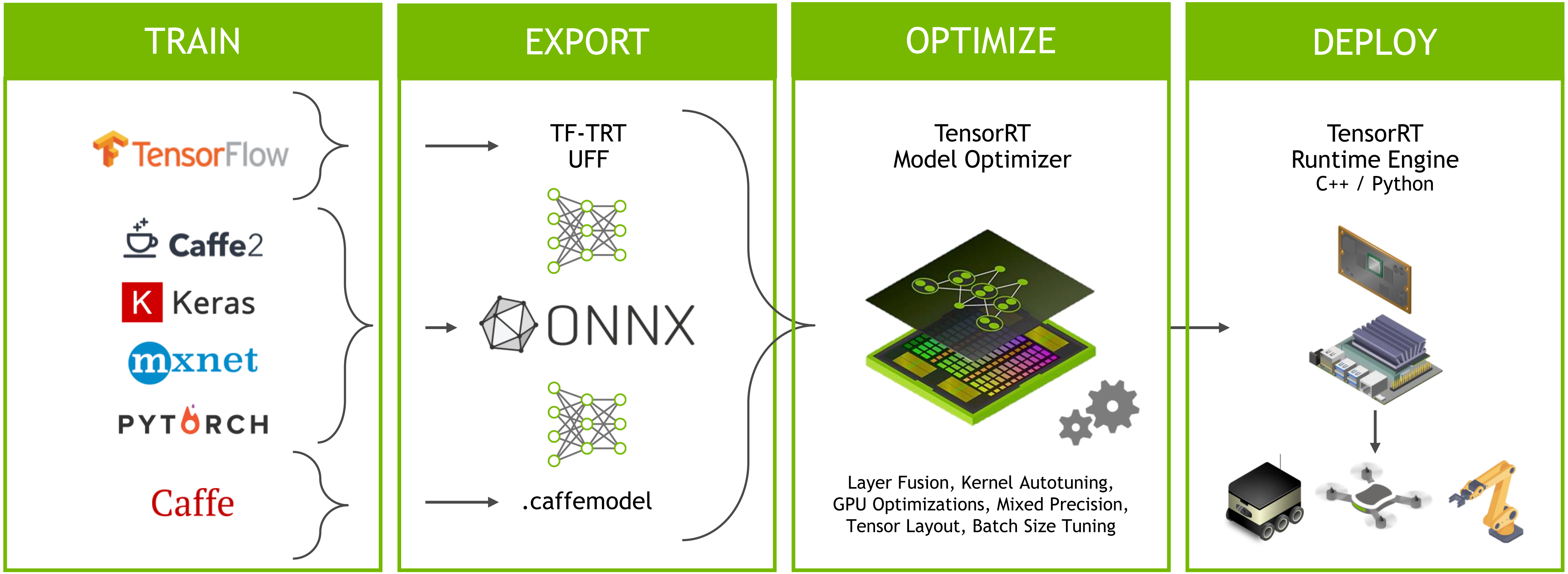
MPI

ROS

JETSON



NVIDIA TensorRT



TRAIN

TensorFlow

Caffe2

Keras

mxnet

PYTORCH

Caffe

EXPORT

TF-TRT
UFF

ONNX

.caffemodel

OPTIMIZE

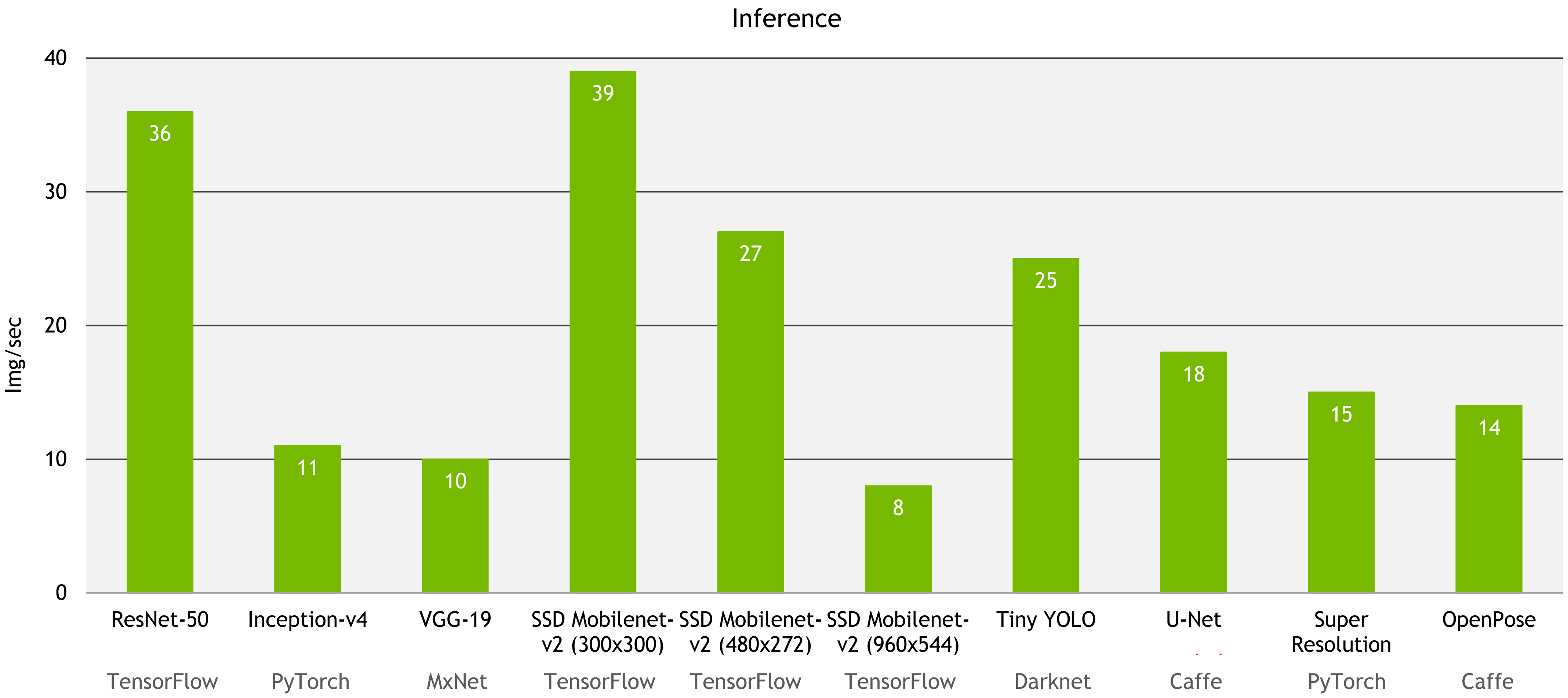
TensorRT
Model Optimizer

Layer Fusion, Kernel Autotuning,
GPU Optimizations, Mixed Precision,
Tensor Layout, Batch Size Tuning

DEPLOY

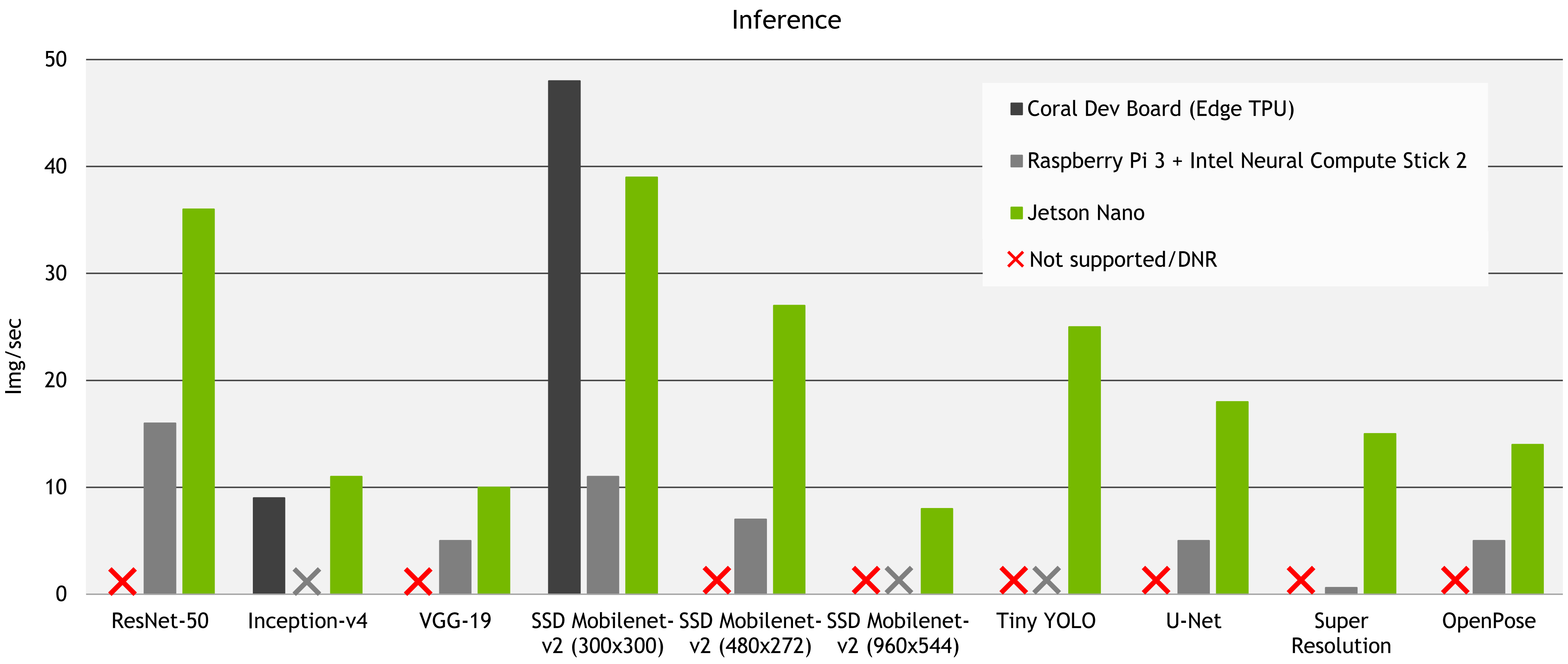
TensorRT
Runtime Engine
C++ / Python

JETSON NANO RUNS MODERN AI

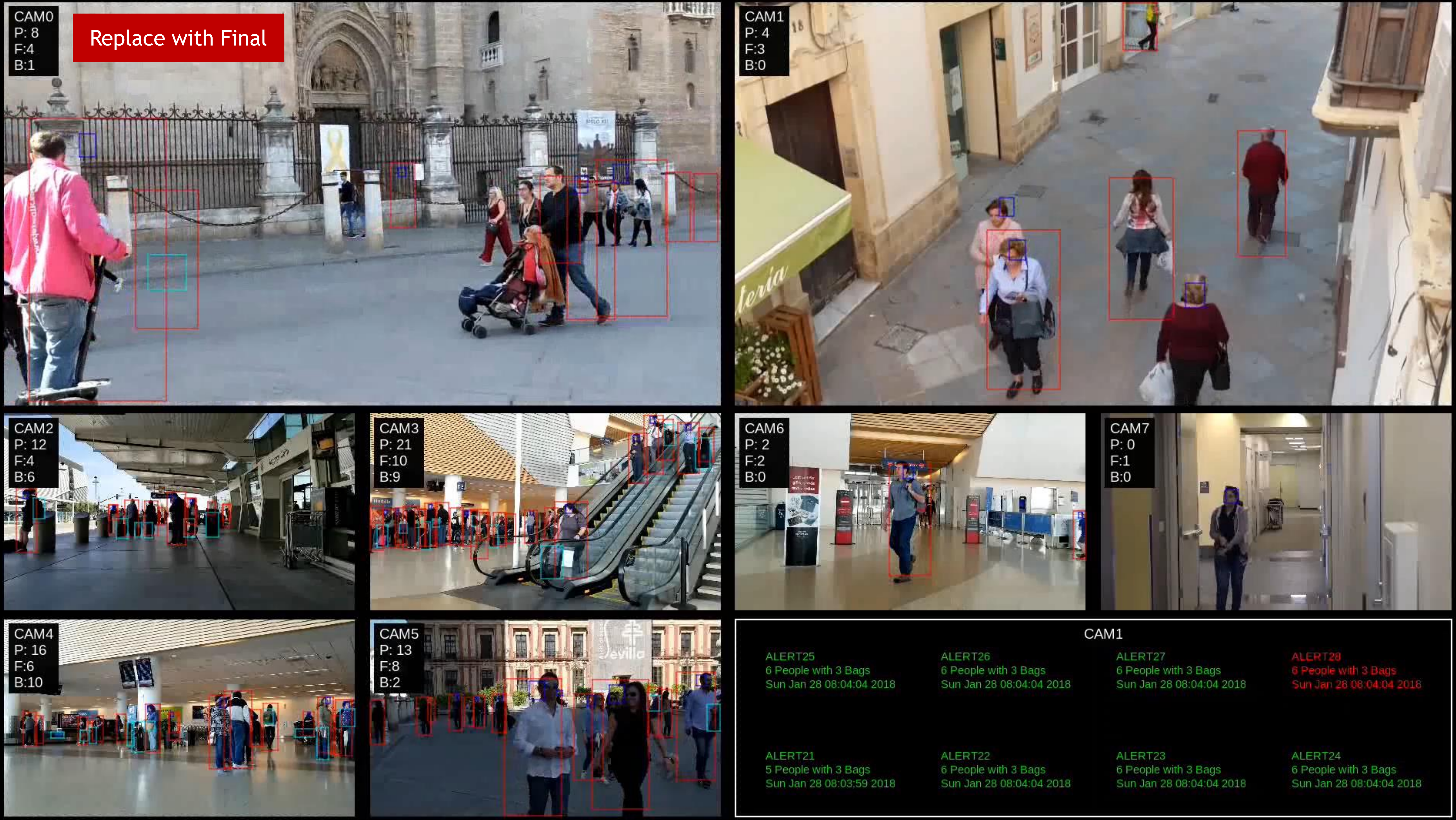


developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks

JETSON NANO RUNS MODERN AI



developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks



CAM1

ALERT25
6 People with 3 Bags
Sun Jan 28 08:04:04 2018

ALERT26
6 People with 3 Bags
Sun Jan 28 08:04:04 2018

ALERT27
6 People with 3 Bags
Sun Jan 28 08:04:04 2018

ALERT28
6 People with 3 Bags
Sun Jan 28 08:04:04 2018

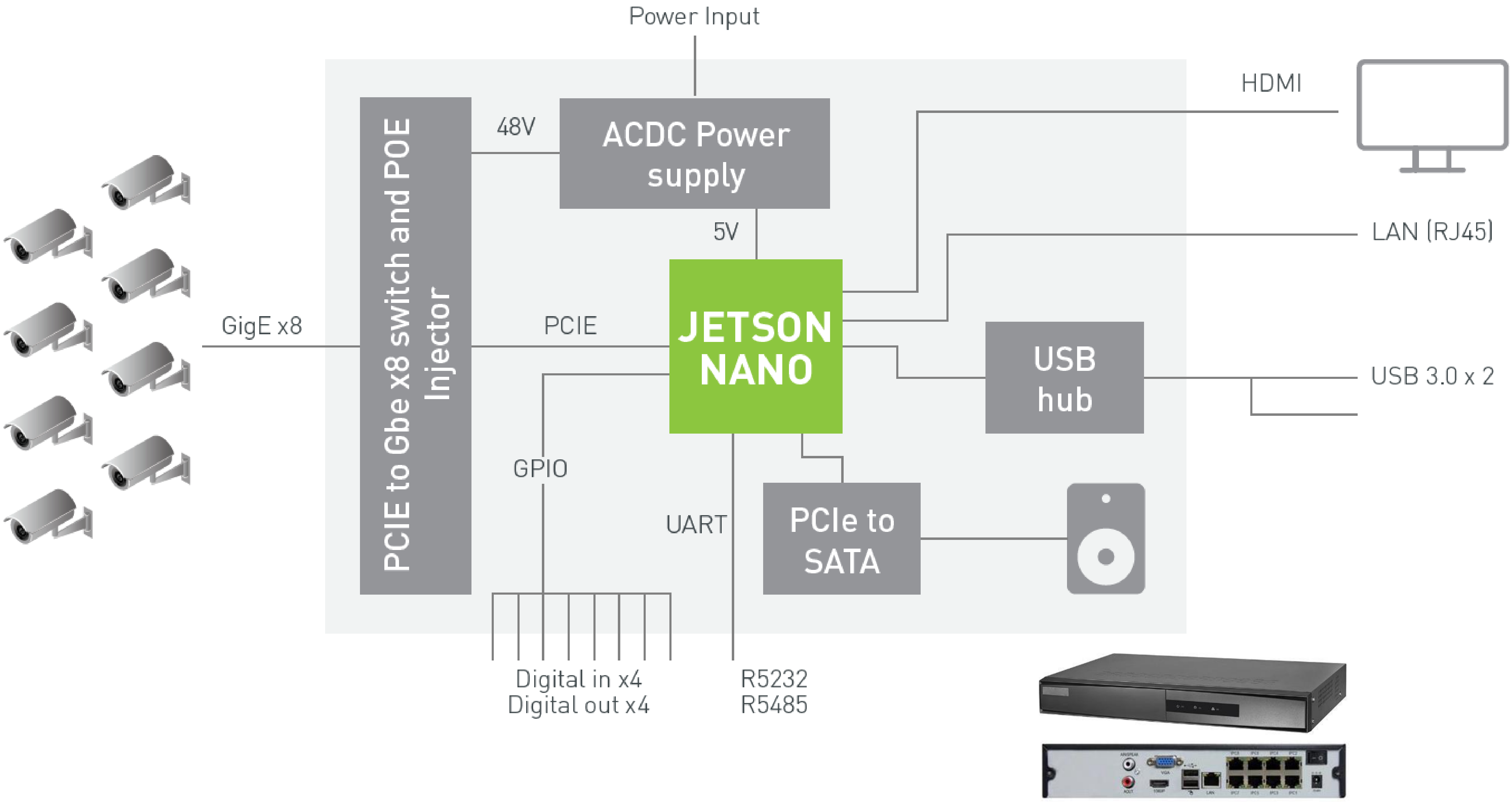
ALERT21
5 People with 3 Bags
Sun Jan 28 08:03:59 2018

ALERT22
6 People with 3 Bags
Sun Jan 28 08:04:04 2018

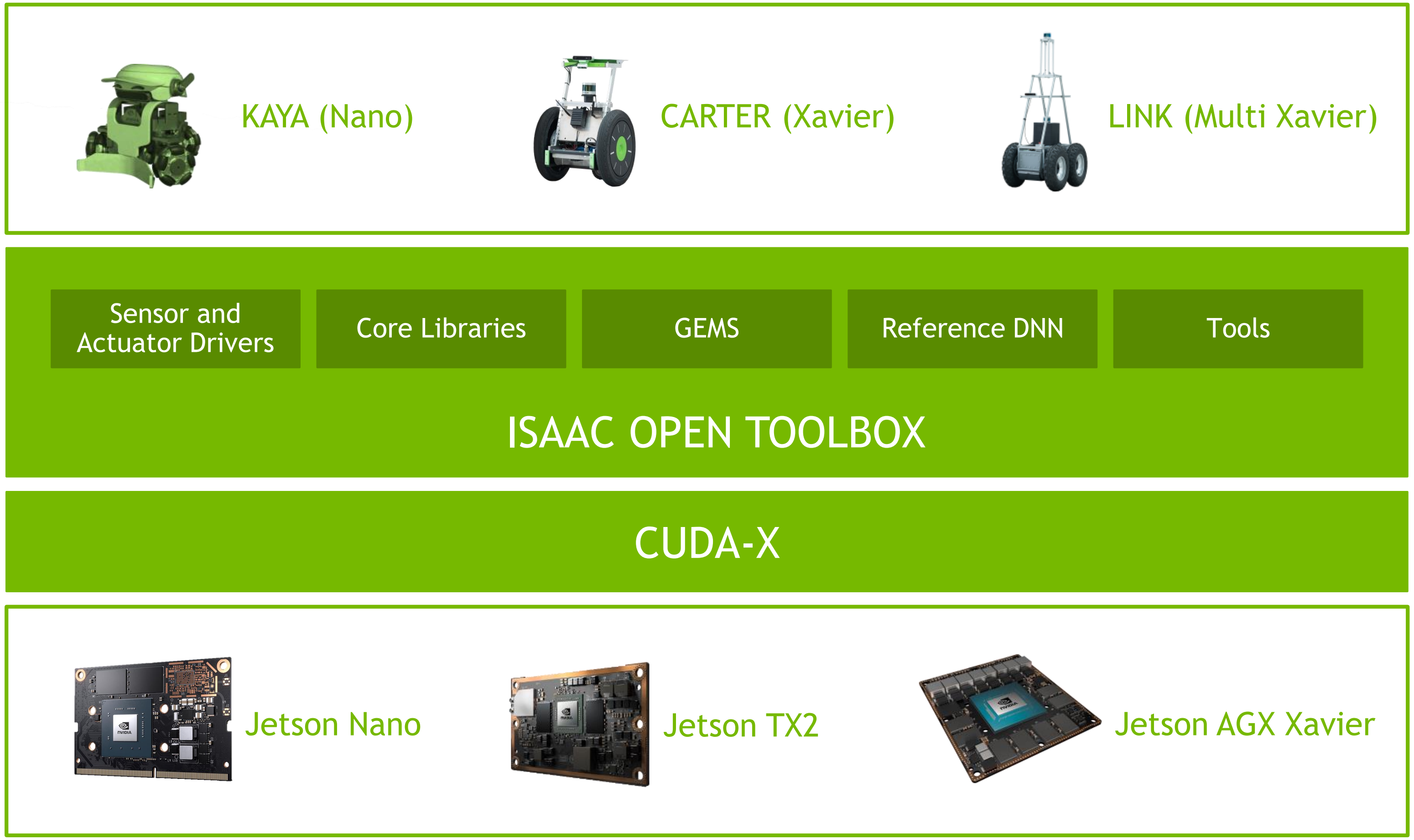
ALERT23
6 People with 3 Bags
Sun Jan 28 08:04:04 2018

ALERT24
6 People with 3 Bags
Sun Jan 28 08:04:04 2018

NETWORK VIDEO RECORDER



ISAAC SDK



KAYA (Nano)

CARTER (Xavier)

LINK (Multi Xavier)

Sensor and Actuator Drivers

Core Libraries

GEMS

Reference DNN

Tools

ISAAC OPEN TOOLBOX

CUDA-X

Jetson Nano

Jetson TX2

Jetson AGX Xavier

Isaac Sim

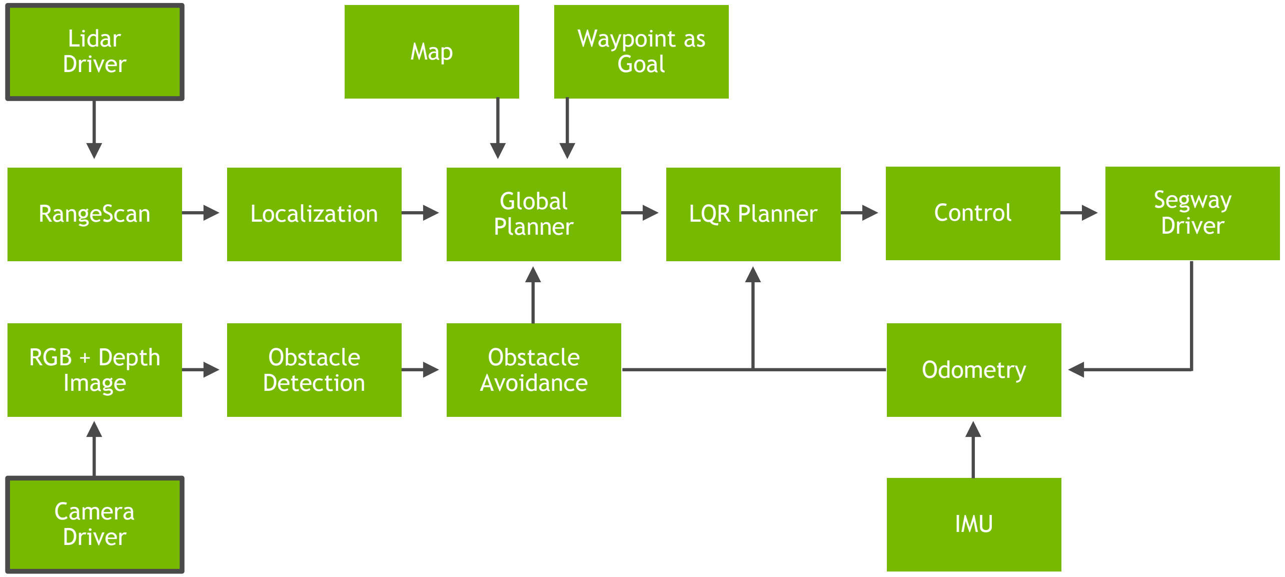
Isaac Gym

developer.nvidia.com/isaac-sdk

ISAAC ROBOTS

NVIDIA Carter

NVIDIA Kaya



Lidar Driver

Map

Waypoint as Goal

RangeScan

Localization

Global Planner

LQR Planner

Control

Segway Driver

RGB + Depth Image

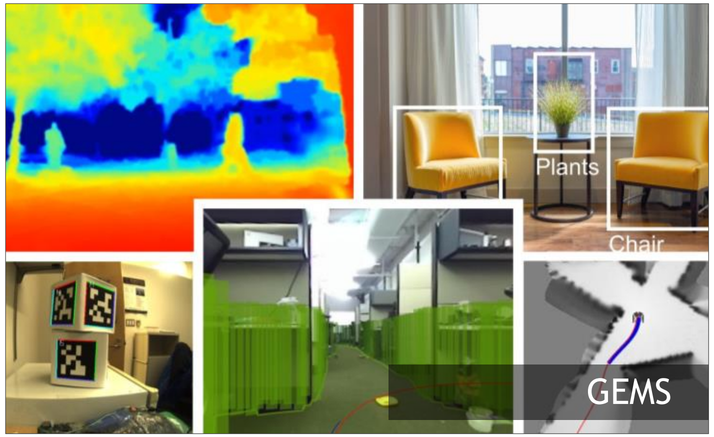
Obstacle Detection

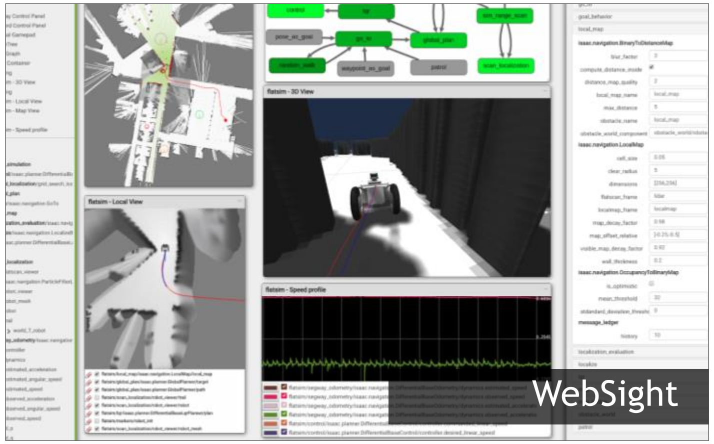
Obstacle Avoidance

Odometry

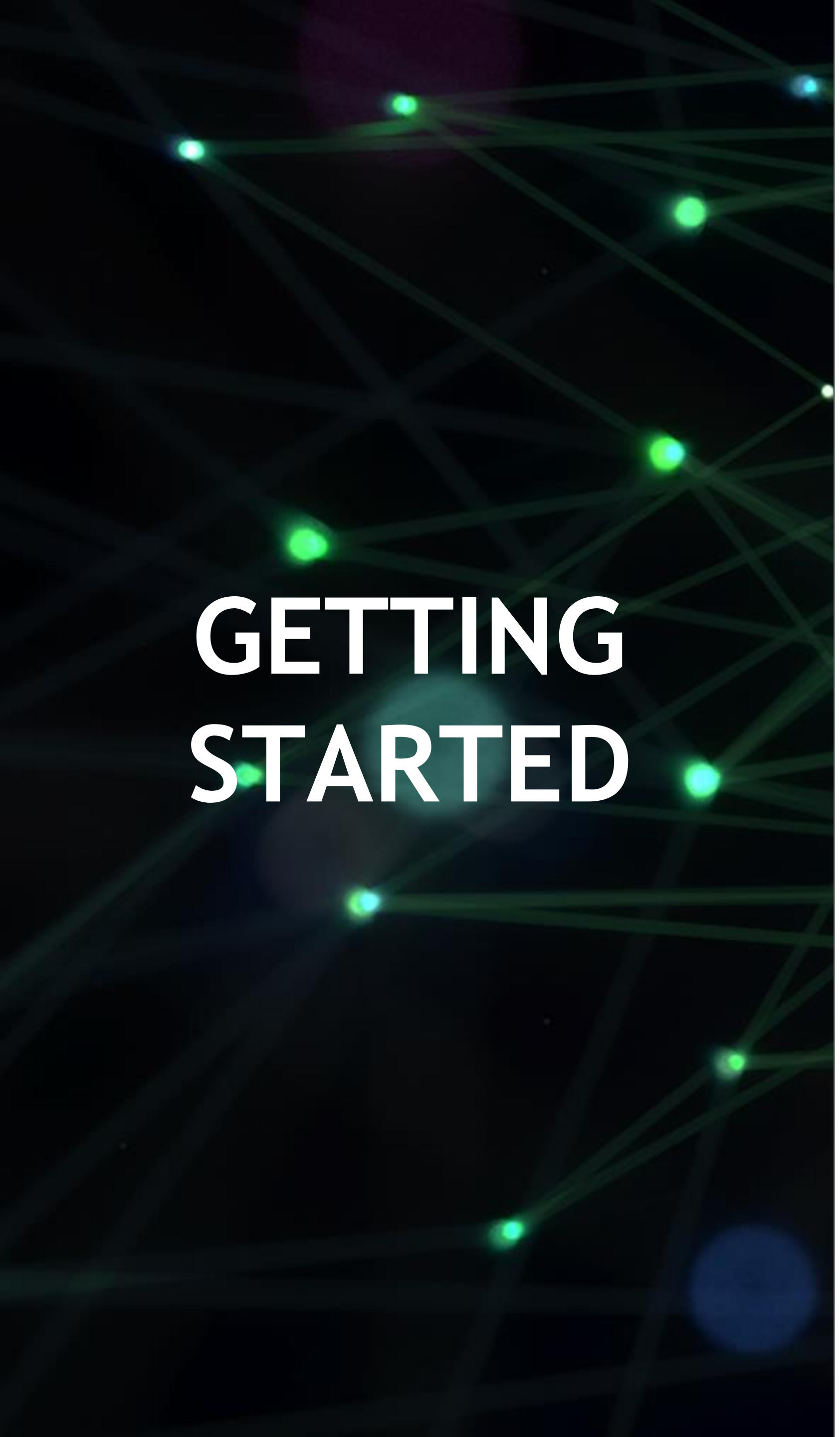
Camera Driver

IMU





developer.nvidia.com/isaac-sdk



GETTING STARTED

Resources

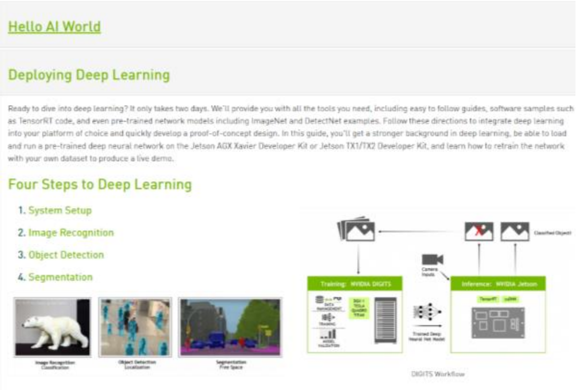
Tutorials

System Setup

Tips and Tricks

Accessories

JETSON NANO RESOURCES



Hello AI World

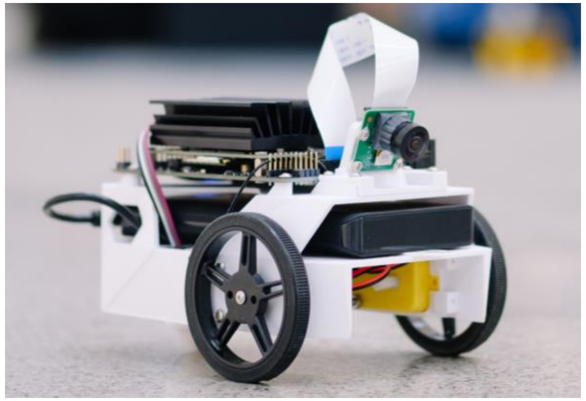
Deploying Deep Learning

Ready to dive into deep learning? It only takes two days. We'll provide you with all the tools you need, including easy to follow guides, software samples such as TensorRT code, and even pre-trained network models including ImageNet and DetectNet examples. Follow these directions to integrate deep learning into your platform of choice and quickly develop a proof-of-concept design. In this guide, you'll get a stronger background in deep learning, be able to load and run a pre-trained deep neural network on the Jetson AGX Xavier Developer Kit or Jetson TX1/TX2 Developer Kit, and learn how to retrain the network with your own dataset to produce a live demo.

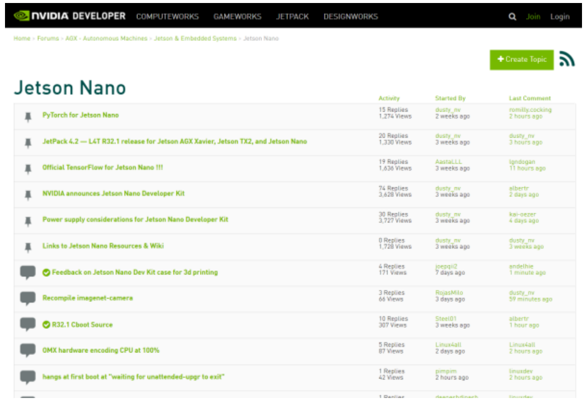
Four Steps to Deep Learning

1. System Setup
2. Image Recognition
3. Object Detection
4. Segmentation

Tutorials



Projects

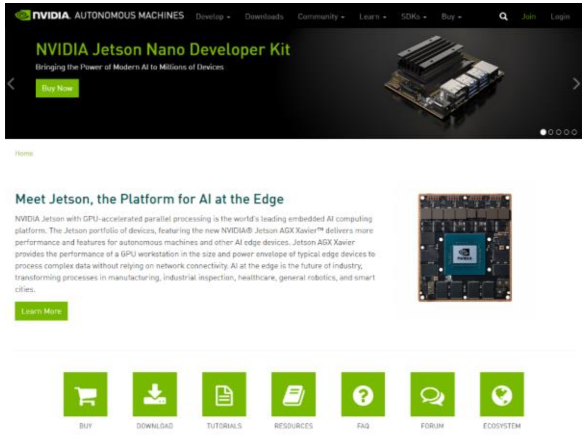


NVIDIA DEVELOPER
COMPUTEWORKS
GAMEWORKS
JETPACK
DESIGNWORKS
Join
Login
Home › Forums › AGX - Autonomous Machines › Jetson & Embedded Systems › Jetson Nano
+ Create Topic

Jetson Nano

	Activity	Started By	Last Comment
PyTorch for Jetson Nano	15 Replies 1,274 Views	dusty_nv 2 weeks ago	romilly.cocking 2 hours ago
JetPack 4.2 — L4T R32.1 release for Jetson AGX Xavier, Jetson TX2, and Jetson Nano	20 Replies 1,330 Views	dusty_nv 3 weeks ago	dusty_nv 3 hours ago
Official TensorFlow for Jetson Nano !!!	19 Replies 1,636 Views	AastaLLL 3 weeks ago	igndogan 11 hours ago
NVIDIA announces Jetson Nano Developer Kit	74 Replies 3,628 Views	dusty_nv 3 weeks ago	albertr 2 days ago
Power supply considerations for Jetson Nano Developer Kit	30 Replies 3,727 Views	dusty_nv 3 weeks ago	kai-oezer 4 days ago
Links to Jetson Nano Resources & Wiki	0 Replies 1,728 Views	dusty_nv 3 weeks ago	dusty_nv 3 weeks ago
✓ Feedback on Jetson Nano Dev Kit case for 3d printing	4 Replies 171 Views	joepqi2 7 days ago	andelhie 1 minute ago
Recompile imagenet-camera	3 Replies 66 Views	RojasMilo 3 days ago	dusty_nv 59 minutes ago
✓ R32.1 Cboot Source	10 Replies 307 Views	Steel01 3 weeks ago	albertr 1 hour ago
OMX hardware encoding CPU at 100%	5 Replies 87 Views	Linux4all 2 days ago	Linux4all 2 hours ago
hangs at first boot at "waiting for unattended-upgr to exit"	1 Replies 42 Views	pimpim 2 hours ago	linuxdev 2 hours ago

Developer Forums



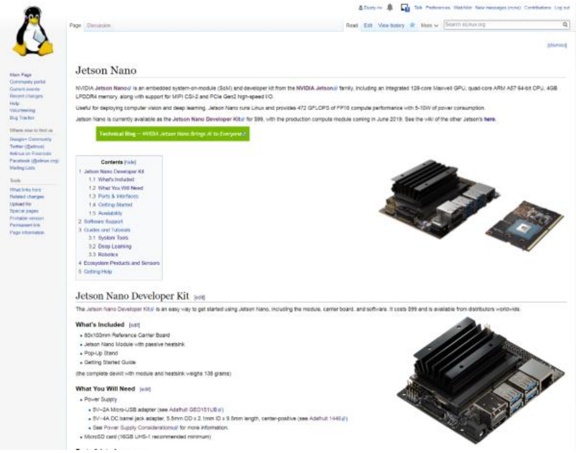
NVIDIA Jetson Nano Developer Kit

Bringing the Power of Modern AI to Millions of Devices

Meet Jetson, the Platform for AI at the Edge

NVIDIA Jetson with GPU-accelerated parallel processing is the world's leading embedded AI computing platform. The Jetson portfolio of devices, featuring the new NVIDIA® Jetson AGX Xavier™ delivers more performance and features for autonomous machines and other AI edge devices. Jetson AGX Xavier provides the performance of a GPU workstation in the size and power envelope of typical edge devices to process complex data without relying on network connectivity. AI at the edge is the future of industry, transforming processes in manufacturing, industrial inspection, healthcare, general robotics, and smart cities.

Jetson Developer Zone



Jetson Nano

NVIDIA Jetson Nano is an embedded system-on-module (SoM) and developer kit from the NVIDIA Jetson family, including an integrated 128-core Maxwell GPU, quad-core ARM A57 64-bit CPU, 4GB LPDDR4 memory, along with support for MIPI CSI-2 and PCIe Gen2 high-speed I/O.

Useful for deploying computer vision and deep learning, Jetson Nano runs Linux and provides 472 GFLOPS of FP16 compute performance with 5-10W of power consumption.

Jetson Nano is currently available as the Jetson Nano Developer Kit for $99, with the production compute module coming in June 2019. See the wiki of the other Jetson's here.

Technical Blog — NVIDIA Jetson Nano Brings AI to Everyone

Contents [hide]

1 Jetson Nano Developer Kit
  1.1 What's Included
  1.2 What You Will Need
  1.3 Ports & Interfaces
  1.4 Getting Started
  1.5 Availability
2 Software Support
3 Guides and Tutorials
  3.1 System Tools
  3.2 Deep Learning
  3.3 Robotics
4 Ecosystem Products and Sensors
5 Getting Help

Jetson Nano Developer Kit [edit]

The Jetson Nano Developer Kit is an easy way to get started using Jetson Nano, including the module, carrier board, and software. It costs $99 and is available from distributors worldwide.

What's Included [edit]

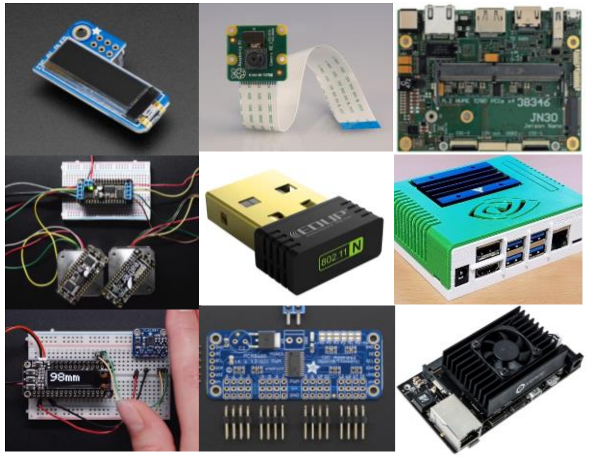
- 80x100mm Reference Carrier Board
- Jetson Nano Module with passive heatsink
- Pop-Up Stand
- Getting Started Guide

(the complete devkit with module and heatsink weighs 138 grams)

What You Will Need [edit]

- Power Supply
  - 5V⎓2A Micro-USB adapter (see Adafruit GEO151UB)
  - 5V⎓4A DC barrel jack adapter, 5.5mm OD x 2.1mm ID x 9.5mm length, center-positive (see Adafruit 1446)
  - See Power Supply Considerations for more information.
- MicroSD card (16GB UHS-1 recommended minimum)

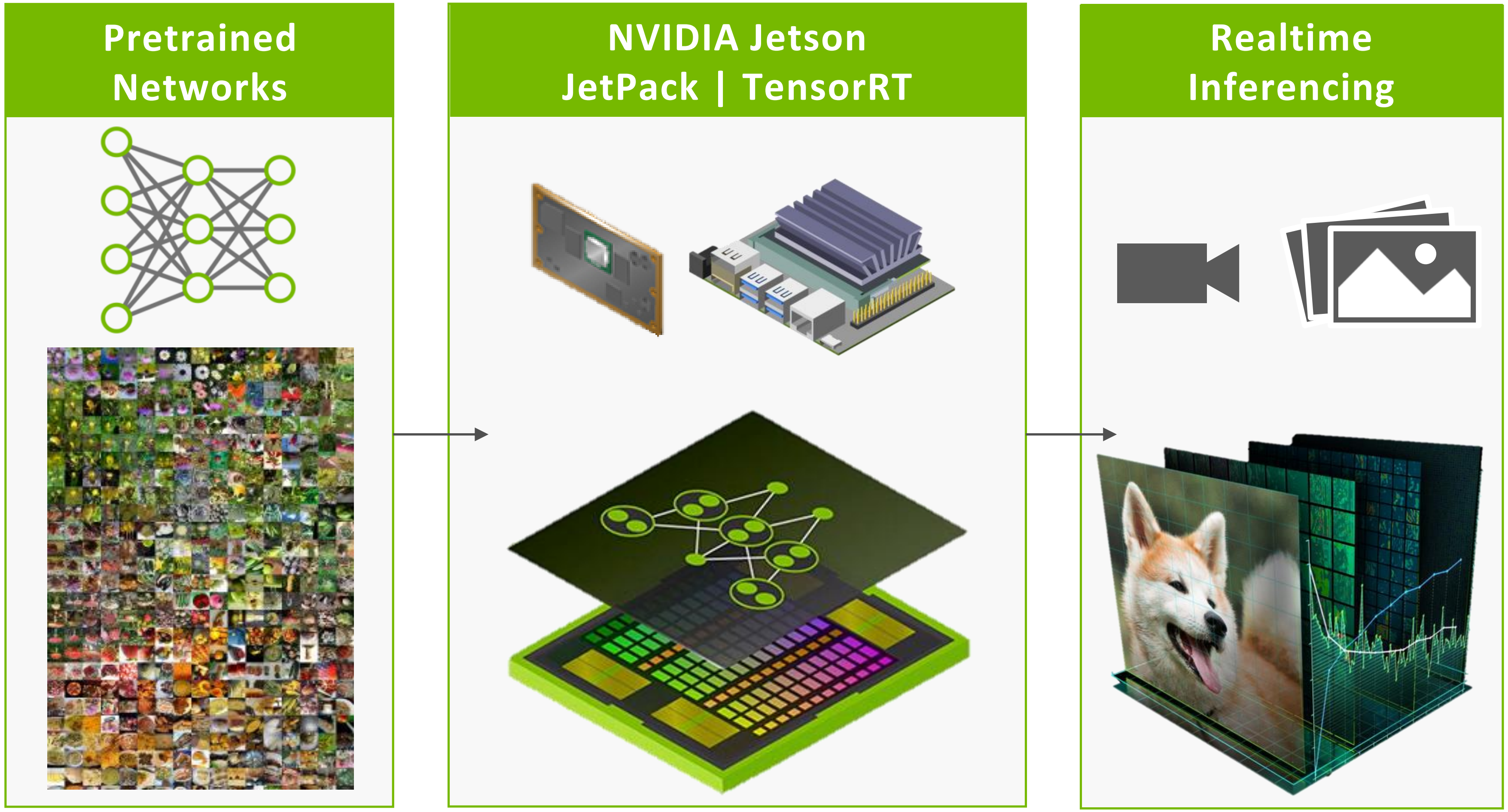
eLinux Wiki



Accessories

HELLO AI WORLD

Getting Started with Deep Learning



Pretrained Networks

NVIDIA Jetson JetPack | TensorRT

Realtime Inferencing

github.com/dusty-nv/jetson-inference

HELLO AI WORLD

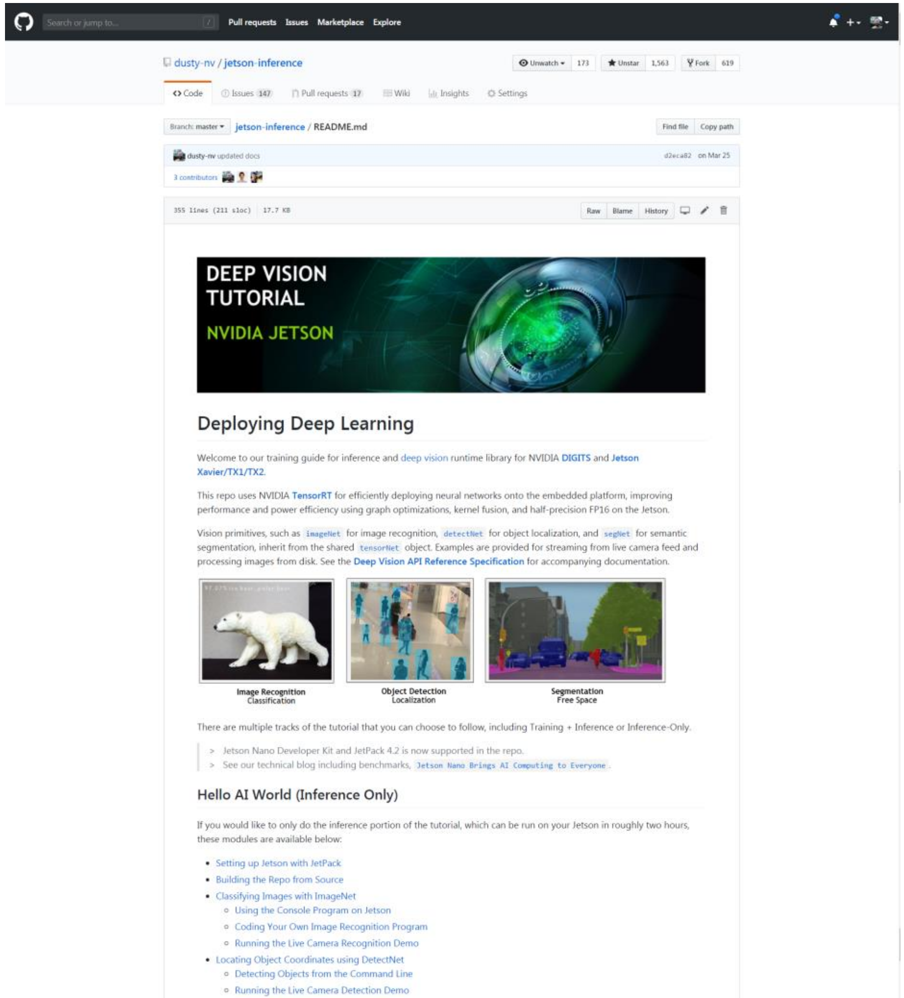
Getting Started with Deep Learning

1. Download and Build the GitHub Repo

```
git clone http://github.com/dusty-nv/jetson-inference
```

2. Classifying Images from Command Line
3. Coding Your Own Recognition Program
4. Realtime Recognition from Live Camera
5. Detecting Objects in Images from Disk
6. Object Detection from Live Camera

github.com/dusty-nv/jetson-inference



Deploying Deep Learning

Welcome to our training guide for inference and deep vision runtime library for NVIDIA **DIGITS** and **Jetson Xavier/TX1/TX2**.

This repo uses NVIDIA **TensorRT** for efficiently deploying neural networks onto the embedded platform, improving performance and power efficiency using graph optimizations, kernel fusion, and half-precision FP16 on the Jetson.

Vision primitives, such as `imageNet` for image recognition, `detectNet` for object localization, and `segNet` for semantic segmentation, inherit from the shared `tensorNet` object. Examples are provided for streaming from live camera feed and processing images from disk. See the **Deep Vision API Reference Specification** for accompanying documentation.

Image Recognition
Classification

Object Detection
Localization

Segmentation
Free Space

There are multiple tracks of the tutorial that you can choose to follow, including Training + Inference or Inference-Only.

> Jetson Nano Developer Kit and JetPack 4.2 is now supported in the repo.
> See our technical blog including benchmarks, `Jetson Nano Brings AI Computing to Everyone`.

Hello AI World (Inference Only)

If you would like to only do the inference portion of the tutorial, which can be run on your Jetson in roughly two hours, these modules are available below:

- Setting up Jetson with JetPack
- Building the Repo from Source
- Classifying Images with ImageNet
  - Using the Console Program on Jetson
  - Coding Your Own Image Recognition Program
  - Running the Live Camera Recognition Demo
- Locating Object Coordinates using DetectNet
  - Detecting Objects from the Command Line
  - Running the Live Camera Detection Demo

HELLO AI WORLD

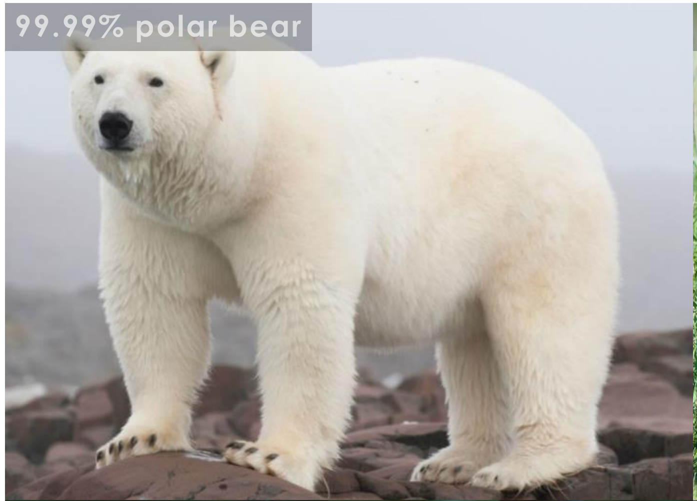
Getting Started with Deep Learning

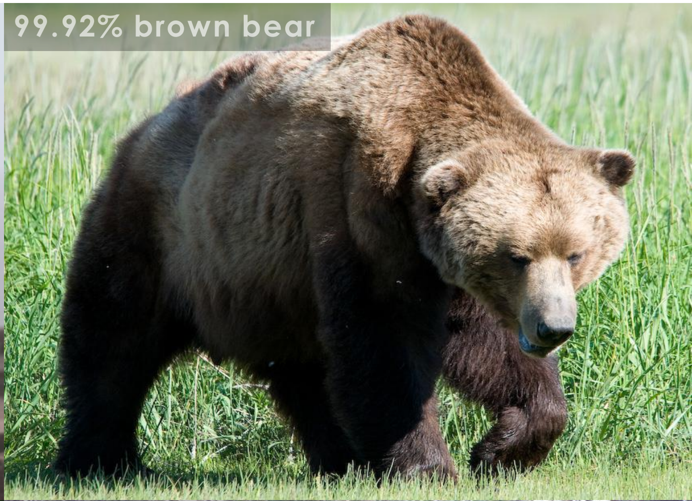
1. Download and Build the GitHub Repo
2. Classifying Images from Command Line

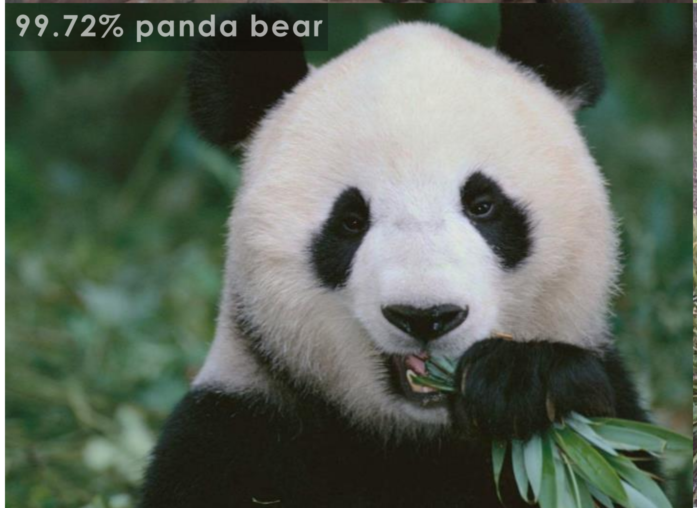
```
./imagenet-console bear_0.jpg output_0.jpg
```

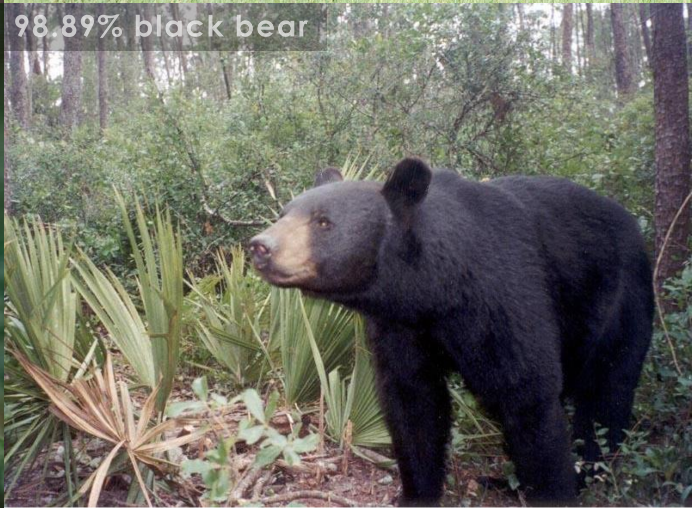
3. Coding Your Own Recognition Program
4. Realtime Recognition from Live Camera
5. Detecting Objects in Images from Disk
6. Object Detection from Live Camera

github.com/dusty-nv/jetson-inference









HELLO AI WORLD

Getting Started with Deep Learning

1. Download and Build the GitHub Repo
2. Classifying Images from Command Line
3. Coding Your Own Recognition Program

   `./my-recognition test-image.jpg`

4. Realtime Recognition from Live Camera
5. Detecting Objects in Images from Disk
6. Object Detection from Live Camera

github.com/dusty-nv/jetson-inference

```
#include <jetson-inference/imageNet.h>
#include <jetson-utils/loadImage.h>

int main( int argc, char** argv )
{
    // load the image recognition network with TensorRT
    imageNet* net = imageNet::Create(imageNet::GOOGLENET);

    // this variable will store the confidence of the classification (between 0 and 1)
    float confidence = 0.0;

    // classify the image with TensorRT on the GPU (hence we use the CUDA pointer)
    // this will return the index of the object class that the image was recognized as
    const int classIndex = net->Classify(imgCUDA, imgWidth, imgHeight, &confidence);

    // make sure a valid classification result was returned
    if( classIndex >= 0 )
    {
        // retrieve the name/description of the object class index
        const char* classDescription = net->GetClassDesc(classIndex);

        // print out the classification results
        printf("image is recognized as '%s' (class #%i) with %f%% confidence\n",
                classDescription, classIndex, confidence * 100.0f);
    }

    // free the network's resources before shutting down
    delete net;
    return 0;
}
```

HELLO AI WORLD

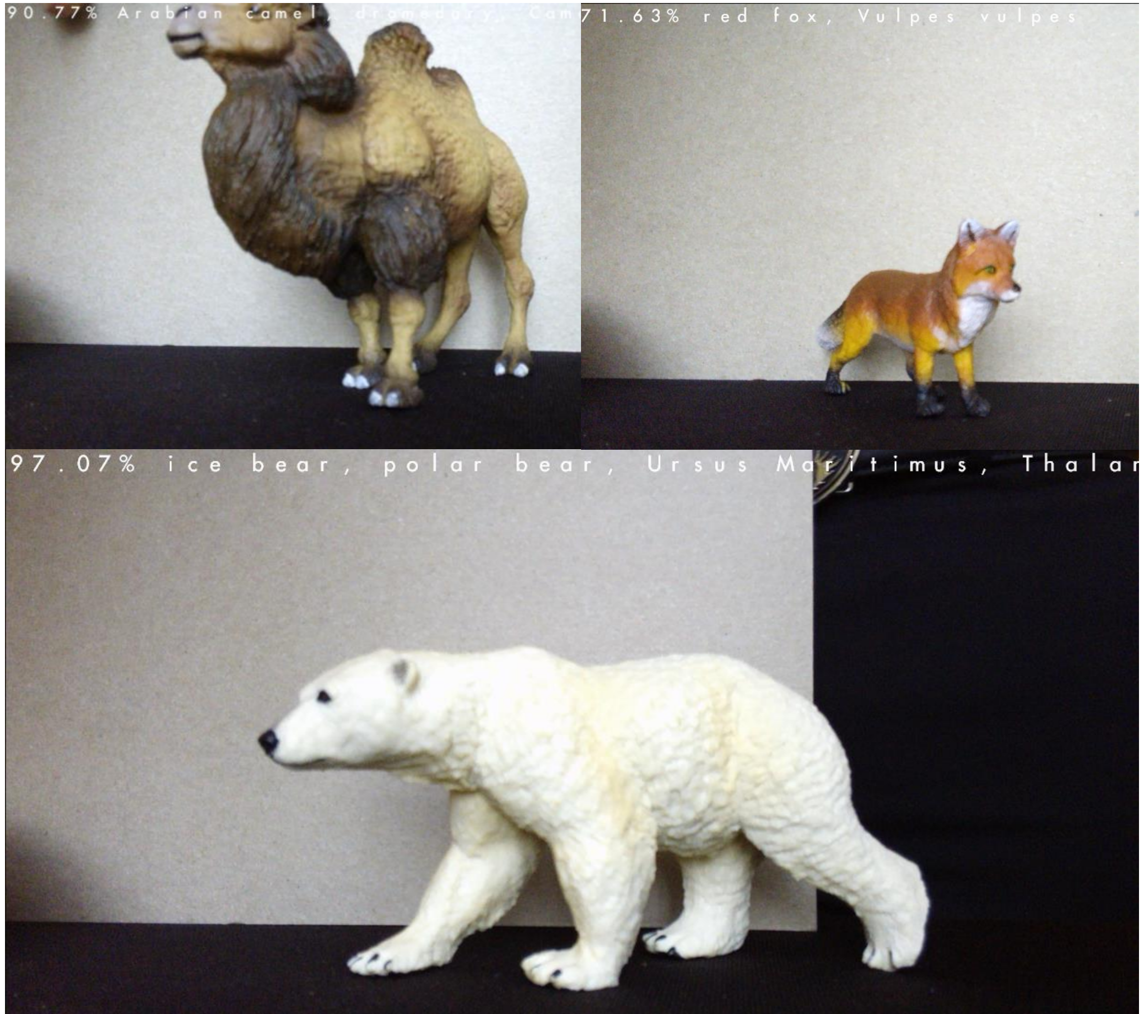
Getting Started with Deep Learning

1. Download and Build the GitHub Repo
2. Classifying Images from Command Line
3. Coding Your Own Recognition Program
4. Realtime Recognition from Live Camera

```
./imagenet-camera googlenet
```

5. Detecting Objects in Images from Disk
6. Object Detection from Live Camera

github.com/dusty-nv/jetson-inference



90.77% Arabian camel, dromedary, Cam

71.63% red fox, Vulpes vulpes

97.07% ice bear, polar bear, Ursus Maritimus, Thalar

HELLO AI WORLD

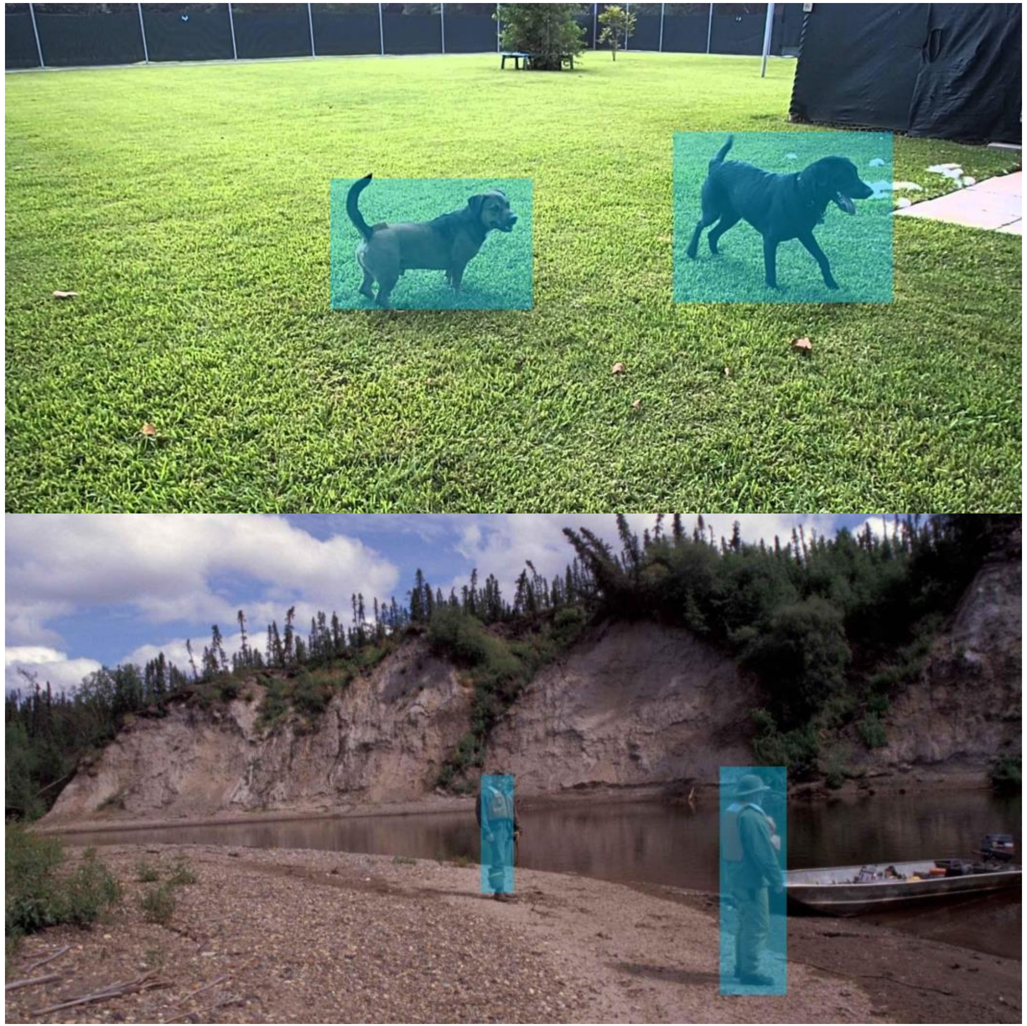
Getting Started with Deep Learning

1. Download and Build the GitHub Repo
2. Classifying Images from Command Line
3. Coding Your Own Recognition Program
4. Realtime Recognition from Live Camera
5. Detecting Objects in Images from Disk

```
./detectnet-console dogs.jpg output.jpg coco-dog
./detectnet-console peds.jpg output.jpg multiped
```

6. Object Detection from Live Camera

github.com/dusty-nv/jetson-inference



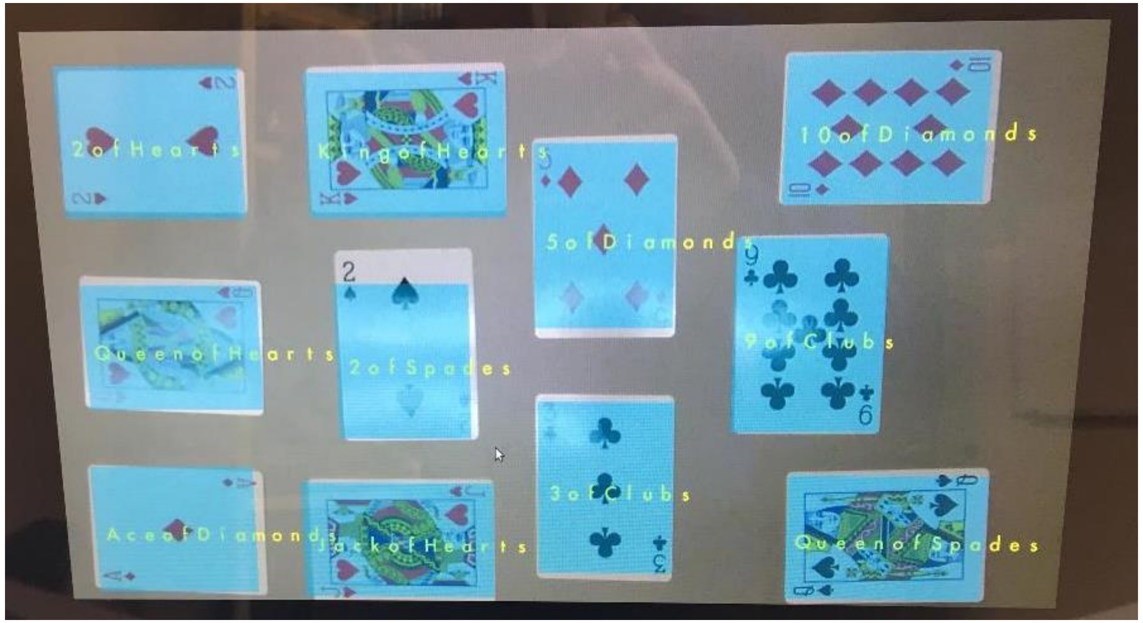
HELLO AI WORLD

Getting Started with Deep Learning

1. Download and Build the GitHub Repo
2. Classifying Images from Command Line
3. Coding Your Own Recognition Program
4. Realtime Recognition from Live Camera
5. Detecting Objects in Images from Disk
6. Object Detection from Live Camera

```
./detectnet-camera <model-name>
```

github.com/dusty-nv/jetson-inference

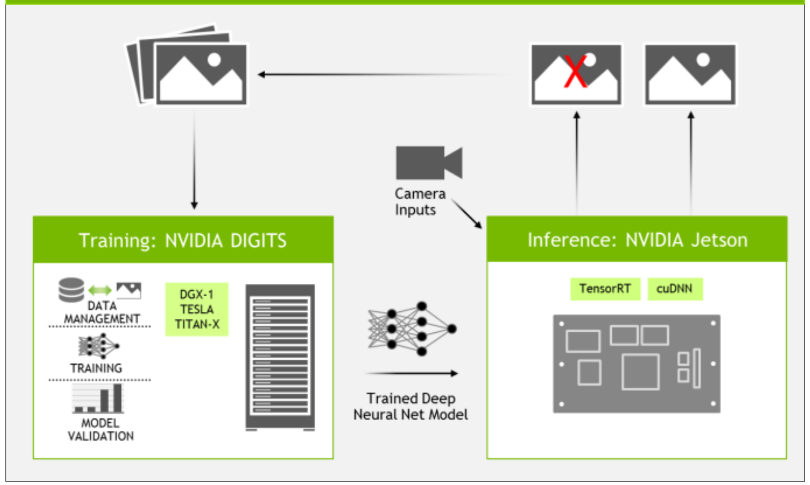


Object Detection Models			
facenet	(faces)	multiped	(humans)
coco-dog	(dogs)	coco-bottle	(bottles)
coco-chair	(chairs)	coco-airplane	(airplanes)

TWO DAYS TO A DEMO

Training + Inference

AI WORKFLOW



Train using DIGITS and cloud/PC
Deploy to the field with Jetson

TRAINING GUIDES

Loading Custom Models on Jetson

Deploying Deep Learning

Locating Object Coordinates using DetectNet

Classifying Images with ImageNet

All the steps required to follow to train
your own models, including the datasets.

DEEP VISION PRIMITIVES

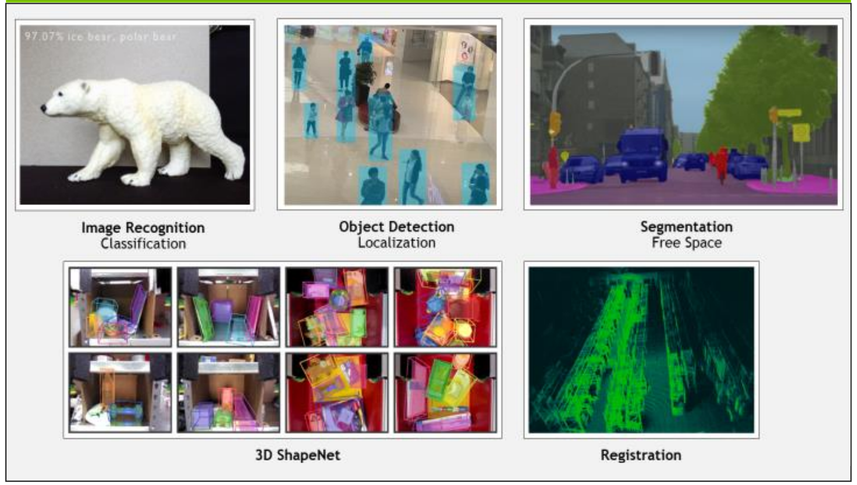
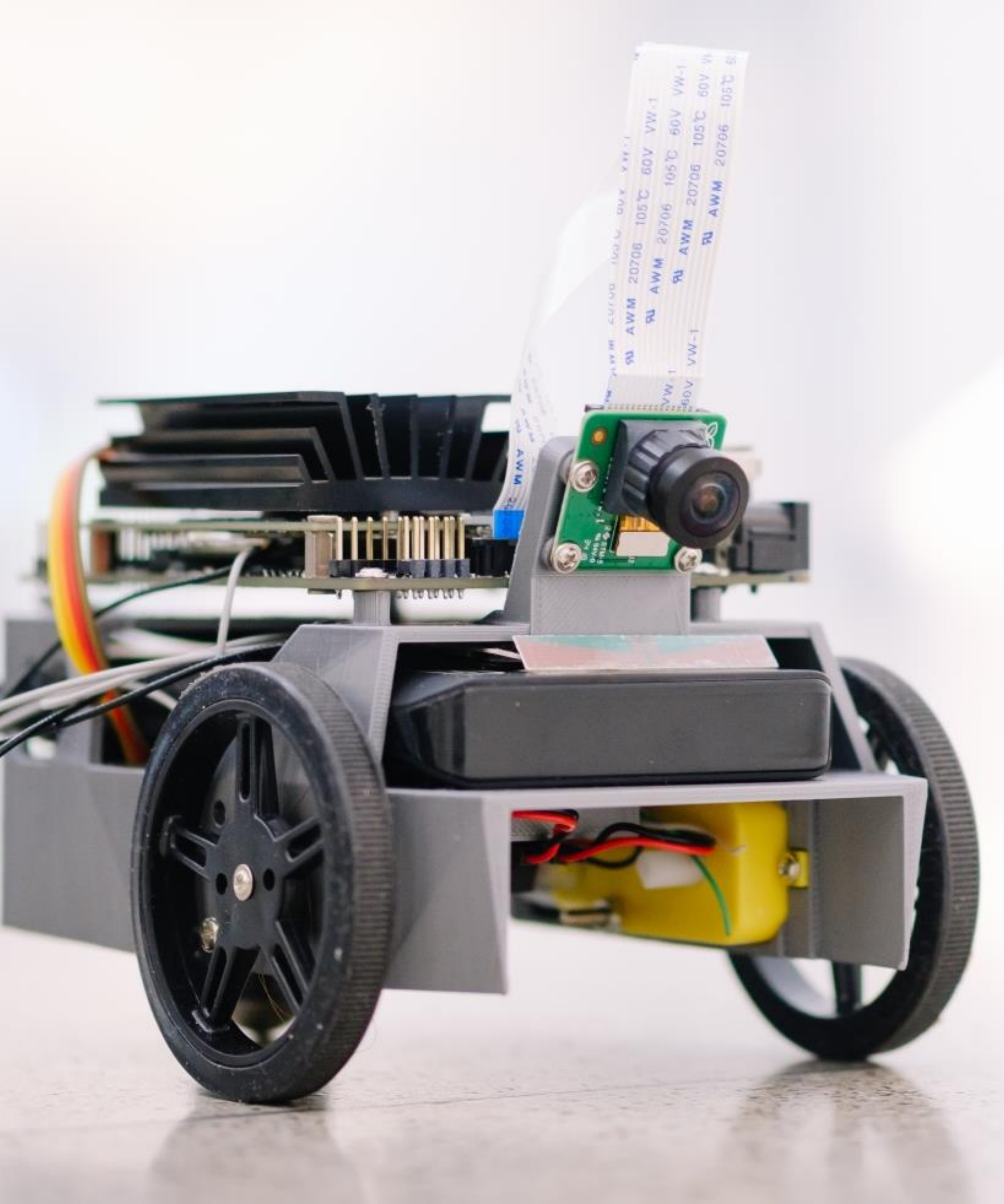


Image Recognition, Object Detection
and Segmentation

github.com/dusty-nv/jetson-inference



JETBOT

~$250 DIY Autonomous Deep Learning Robotics Kit

Programmable through Jupyter IPython Notebooks

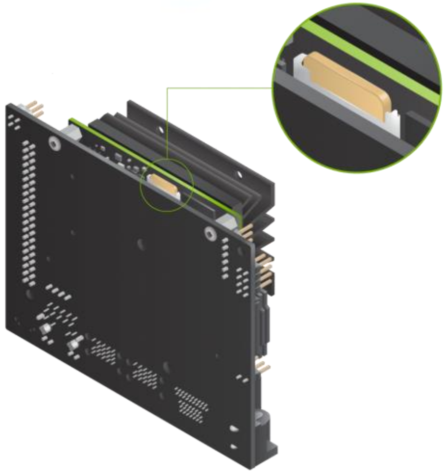
Trainable DNNs for obstacle detection, object following, path planning, and navigation

ROS support and Gazebo simulator available

Join our upcoming JetBot webinar, May 16 2019

github.com/NVIDIA-AI-IOT/JetBot

GPU TECHNOLOGY CONFERENCE
SAN JOSE McENERY CONVENTION CENTER



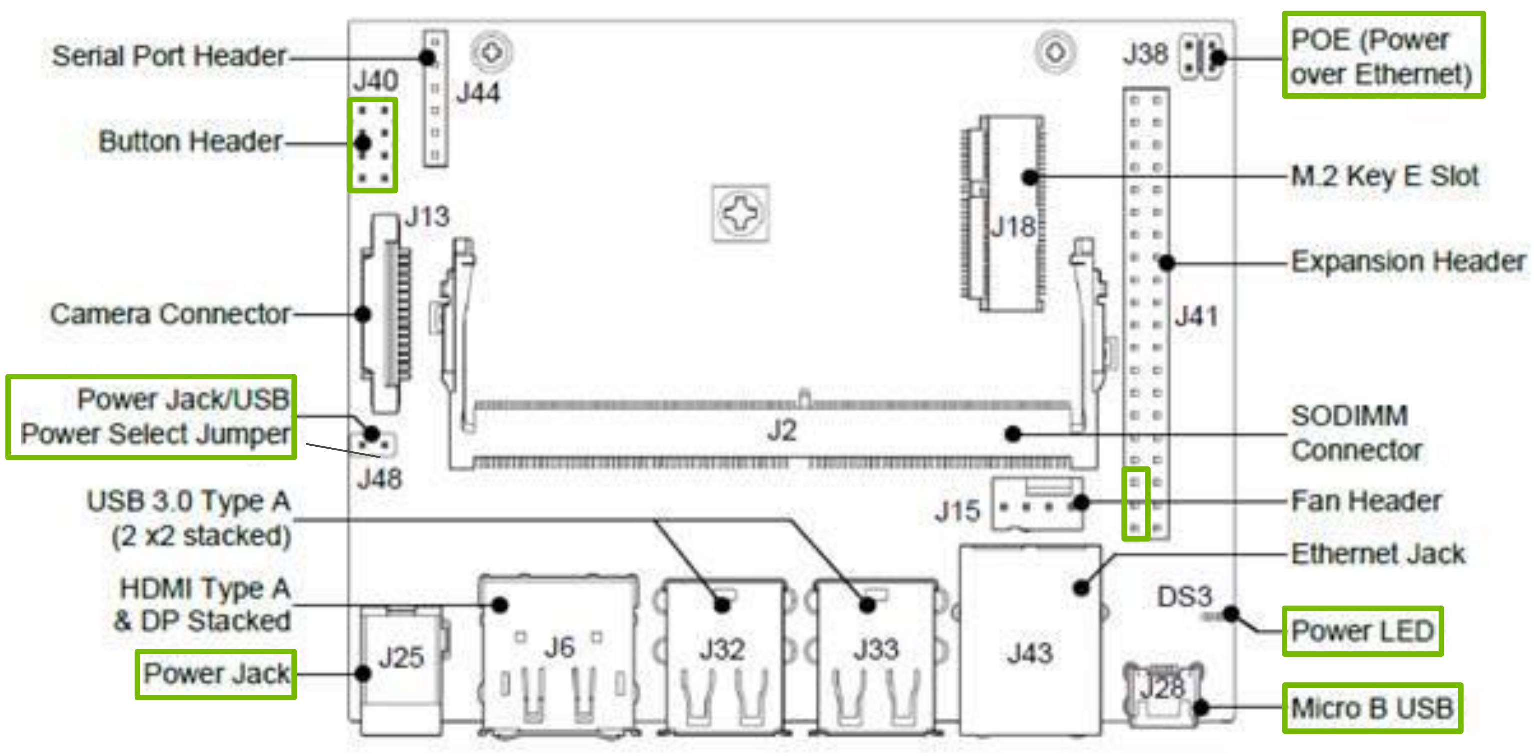
SYSTEM SETUP

- Device is booted from a MicroSD card
  - 16GB UHS-1 recommended minimum
- Download the SD card image from NVIDIA.com
- Flash the SD card image with Etcher program
  - From a Windows/Mac/Linux PC
  - You can also flash JetPack with NV SDK Manager
- Insert the MicroSD card into the slot located on the underside of the Jetson Nano module
- Connect keyboard, mouse, display, and power supply
- Board will automatically boot when power is applied
  - Green power LED will light

NVIDIA.com/JetsonNano-Start

POWER SUPPLIES

- 5V⎓2A Micro-USB charger
  - Adafruit #1995
- 5V⎓4A DC barrel jack adapter
  - Adafruit #1466
  - 5.5mm OD x 2.1mm ID x 9.5mm length
  - Place a jumper on header J48
- J41 Expansion Header, pins 2/4
  - Up to 5V⎓3A per pin (5V⎓6A total)
- Power over Ethernet (PoE)
  - Standard PoE supply is 48V
  - Use a PoE hat or 5V regulator



- J40 Button Header can disable Auto Power-On
  - Manual Power-On / Reset
  - Enter Recovery Mode

POWER MODES

Different power mode presets: 5W and 10W

Default mode is 10W

Users can create their own presets, specifying clocks and online cores in `/etc/nvpmodel.conf`

```
< POWER_MODEL ID=1 NAME=5W >
  CPU_ONLINE CORE_0 1
  CPU_ONLINE CORE_1 1
  CPU_ONLINE CORE_2 0
  CPU_ONLINE CORE_3 0
  CPU_A57 MAX_FREQ 918000
  GPU MAX_FREQ 640000000
  EMC MAX_FREQ 1600000000
```

Power Mode	10W†	5W
Mode ID	0	1
Online CPU Cores	4	2
CPU Max Frequency (MHz)	1428	918*
GPU Max Frequency (MHz)	921	640*
Memory Max Freq. (MHz)	1600	1600

† Default Mode is 10W (ID:0)

* Rounded at runtime to closest discrete freq. available

NVIDIA Power Model Tool

```
sudo nvpmodel -q     (for checking the active mode)

sudo nvpmodel -m 0   (for changing mode, persists after reboot)

sudo jetson_clocks   (to disable DVFS and lock clocks to max for active mode)
```

PERFORMANCE MONITOR

Run `sudo tegrastats` to launch the performance/utilization monitor:

```
RAM 1216/3963MB (lfb 330x4MB) IRAM 0/252kB(lfb 252kB)
CPU [27%@102,36%@307,6%@204,35%@518] EMC_FREQ 19%@204 GR3D_FREQ 0%@76 APE 25
PLL@25C CPU@29.5C PMIC@100C GPU@27C AO@34C thermal@28C POM_5V_IN 1532/1452
POM_5V_GPU 0/20 POM_5V_CPU 241/201
```

Memory	Memory Used / Total Capacity	CPU	Utilization / Frequency (MHz)
Memory	Bandwidth % @ Frequency (MHz)	GPU	Utilization / Frequency (MHz)
Thermal	Zone @ Temperature (°C)	Power	Current Consumption (mW) / Average (mW)

Refer to the L4T Developer Guide for more options and documentation on the output.

docs.nvidia.com/jetson

USING GPIO

- Similar 40-pin header to rPI, 3.3V logic levels
- Adafruit Blinka + SeeedStudio Grove support
- Jetson.GPIO Python library
  - Compatible API with rPI.GPIO
  - Docs & samples in `/opt/nvidia/jetson-gpio/`
- sysfs I/O access from `/sys/class/gpio/`
  - **Map GPIO pin** `echo 38 > /sys/class/gpio/export`
  - **Set direction** `echo out > /sys/class/gpio/gpio38/direction`
  - **Bit-banging** `echo 1 > /sys/class/gpio/gpio38/value`
  - **Unmap GPIO** `echo 38 > /sys/class/gpio/unexport`
  - **Query status** `cat /sys/kernel/debug/gpio`
  - https://www.kernel.org/doc/Documentation/gpio/sysfs.txt
- C/C++ programs (and other languages) can use same sysfs files
- I²C – `libi2c` for C/C++ and Python

J41 Expansion Header

sysfs GPIO	Name	Pin	Pin	Name	sysfs GPIO
	3.3V	1	2	5.0V	
	I2C_2_SDA	3	4	5.0V	
	I2C_2_SCL	5	6	GND	
gpio216	AUDIO_MCLK	7	8	UART_2_TX	
	GND	9	10	UART_2_RX	
gpio50	UART_2_RTS	11	12	I2S_4_SCLK	gpio79
gpio14	SPI_2_SCK	13	14	GND	
gpio194	LCD_TE	15	16	SPI_2_CS1	gpio232
	3.3V	17	18	SPI_2_CS0	gpio15
gpio16	SPI_1_MOSI	19	20	GND	
gpio17	SPI_1_MISO	21	22	SPI_2_MISO	gpio13
gpio18	SPI_1_SCK	23	24	SPI_1_CS0	gpio19
	GND	25	26	SPI_1_CS1	gpio20
	I2C_1_SDA	27	28	I2C_1_SCL	
gpio149	CAM_AF_EN	29	30	GND	
gpio200	GPIO_PZ0	31	32	LCD_BL_PWM	gpio168
gpio38	GPIO_PE6	33	34	GND	
gpio76	I2S_4_LRCK	35	36	UART_2_CTS	gpio51
gpio12	SPI_2_MOSI	37	38	I2S_4_SDIN	gpio77
	GND	39	40	I2S_4_SDOUT	gpio78

JETSON NANO ACCESSORIES

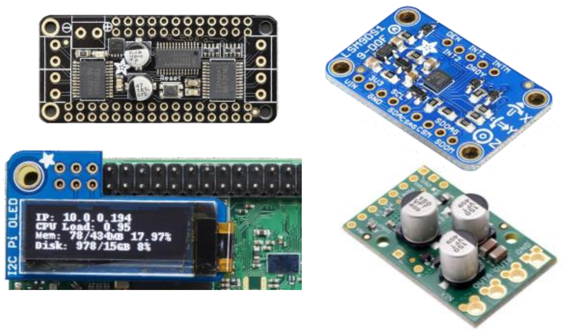
Printable Enclosures

Battery Packs

Carriers

5V Fans

GPIO Hats



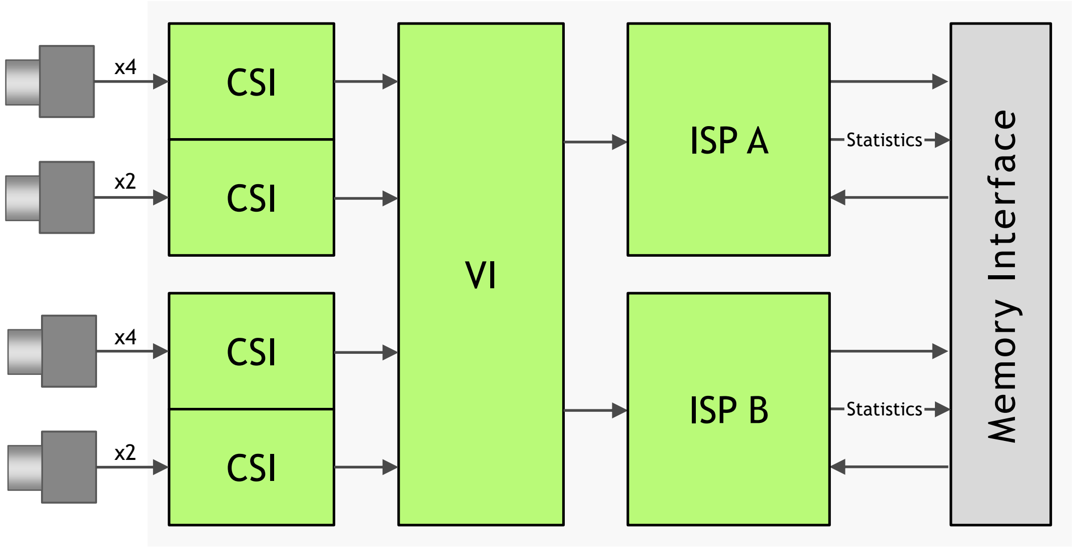
Sensors & Cameras

eLinux.org/Jetson_Nano

CAMERA CAPTURE

- NVIDIA Argus (libargus)
  - Low-overhead offloaded ingest & ISP for MIPI CSI sensors
  - Docs & samples in `/usr/src/tegra_multimedia_api/argus/`
  - `argus_camera` – C++/Python wrapper library on [GitHub]()
- GStreamer
  - `nvarguscamerasrc` element uses Argus internally

  -
    ```
    gst-launch-1.0 nvarguscamerasrc ! ‘video/x-raw(memory:NVMM), \
    width=(int)1920, height=(int)1080, format=(string)NV12, \
    framerate=(fraction)30/1' ! nvoverlaysink -e
    ```
  - `nvgstcapture` camera viewer application
- V4L2
  - Interface with USB cameras and MIPI CSI YUV sensors (`/dev/video`)
  - `libv4l` (C/C++), `pip install v4l2` (Python), `v4l2src` (GStreamer)
  - https://www.kernel.org/doc/html/v4.9/media/uapi/v4l/v4l2.html



Up to three MIPI CSI-2 x4 cameras or four cameras in x4/x2 configurations
(12 MIPI CSI-2 lanes total)

VIDEO CODECS

- Multi-stream HW encoder and decoder engines
- GStreamer
  - NV Encoder elements: `omxh265enc`, `omxh264enc`, ect.

  -
    ```
    gst-launch-1.0 videotestsrc ! 'video/x-raw, format=(string)I420, \
    width=(int)1920, height=(int)1080' ! omxh265enc ! matroskamux ! \
    filesink location=test.mkv -e
    ```
  - NV Decoder elements: `omxh265dec`, `omxh264dec`, ect.

  -
    ```
    gst-launch-1.0 filesrc location=test.mkv ! matroskademux ! \
    h265parse ! omxh265dec ! nvoverlaysink -e
    ```
  - More pipelines in L4T Accelerated GStreamer User Guide
- V4L2 Extensions
  - NV Encoder: `/dev/nvhost-msenc` (YUV in, H.264/H.265 out)
  - NV Decoder: `/dev/nvhost-nvdec` (Bitstream in, NV12/YUV out)
  - Documentation + samples included with L4T Multimedia API

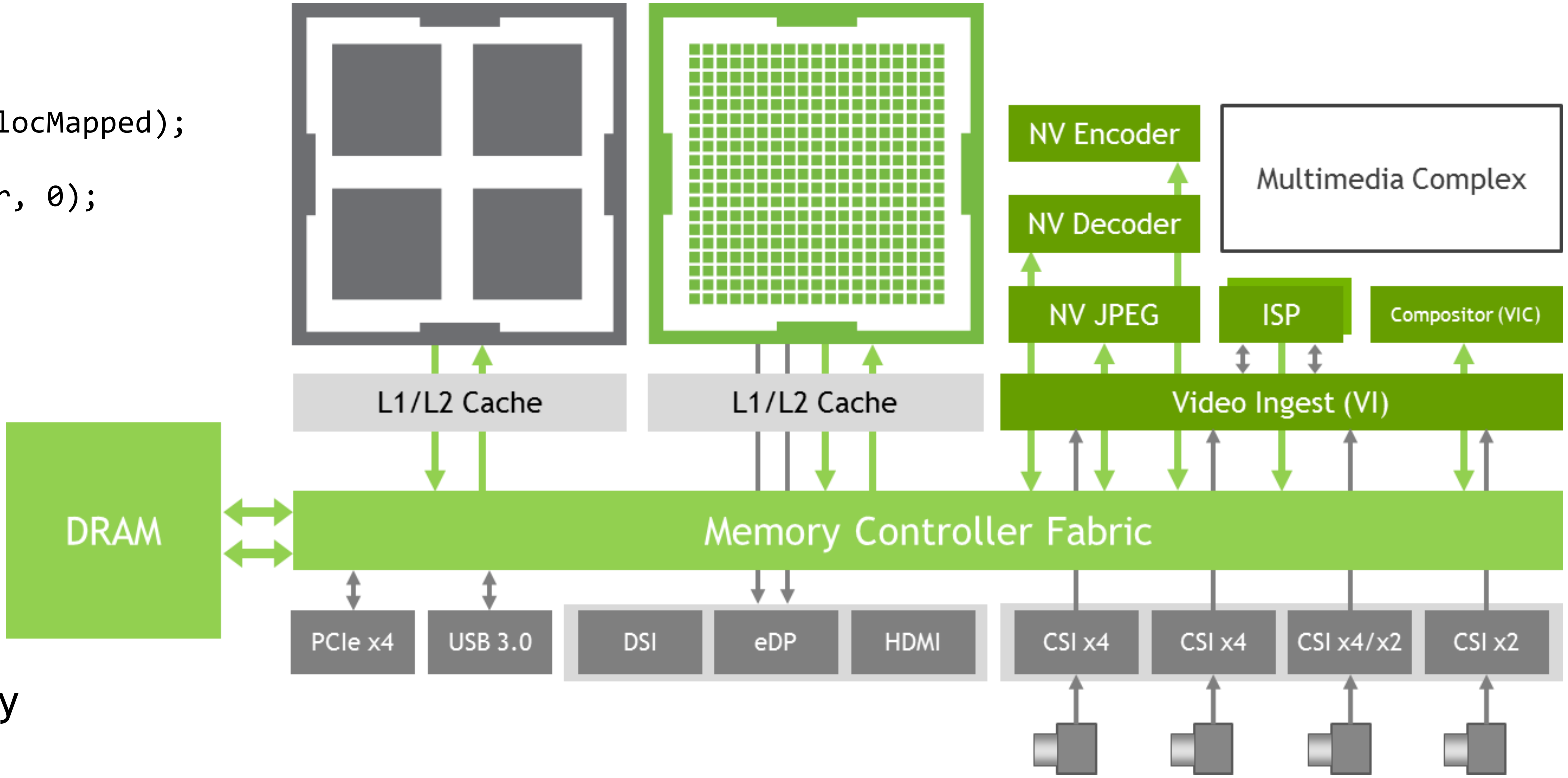
Encoder Profile	
H.265 (Main, Main 10)	4Kp30 | (2x) 1080p60 | (4x) 1080p30
H.264 (Base, Main, High)	4Kp30 | (2x) 1080p60 | (4x) 1080p30
H.264 (MVC Stereo)	1440p30 | 1080p60 | (2x) 1080p30
VP8	4Kp30 | (2x) 1080p60 | (4x) 1080p30
JPEG	600 MP/s

Decoder Profile	
H.265 (Main, Main 10)	4Kp60 | (2x) 4Kp30 | (4x) 1080p60 | (8x) 1080p30
H.264 (Base, Main, High)	4Kp60 | (2x) 4Kp30 | (4x) 1080p60 | (8x) 1080p30
H.264 (MVC Stereo)	4Kp30 | (2x) 1080p60 | (4x) 1080p30
VP9 (Profile 0, 8-bit)	4Kp60 | (2x) 4Kp30 | (4x) 1080p60 | (8x) 1080p30
VP8	4Kp60 | (2x) 4Kp30 | (4x) 1080p60 | (8x) 1080p30
VC-1 (Simple, Main, Adv.)	(2x) 1080p60* | (4x) 1080p30*
MPEG-2 (Main)	4Kp60 | (2x) 4Kp30 | (4x) 1080p60* | (8x) 1080p30*
JPEG	600 MP/s

* Supports progressive and interlaced formats

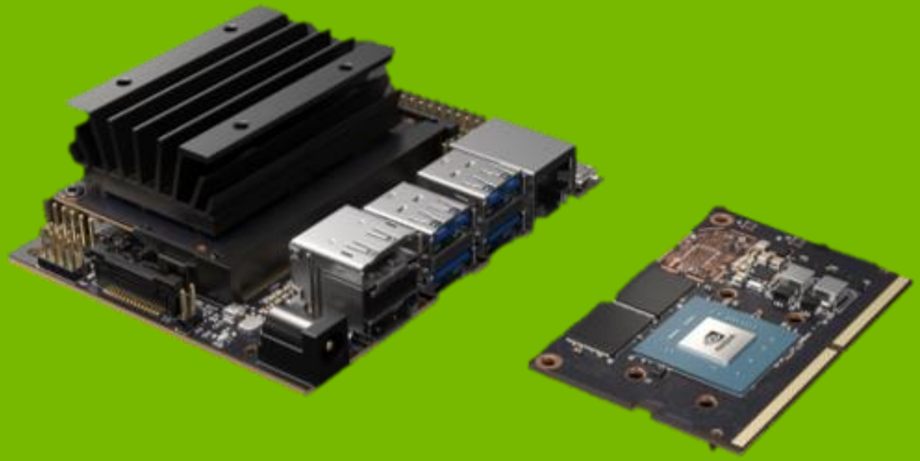
ZERO COPY

- Shared memory fabric allows processor engines to access the same memory, without needing to copy between them
- CUDA Mapped Memory API's
  - `cudaHostAlloc(&cpuPtr, size, cudaHostAllocMapped);`
  - `cudaHostGetDevicePointer(&gpuPtr, cpuPtr, 0);`
  - No `cudaMemcpy()` required
- CUDA Unified Memory
  - `cudaMallocManaged()`
  - Coherent synchronization and caching
  - Disregards data movement on Jetson
- EGLStreams – graphics API interoperability
- Argus, NV V4L2 extensions, and DeepStream libraries are optimized for using ZeroCopy



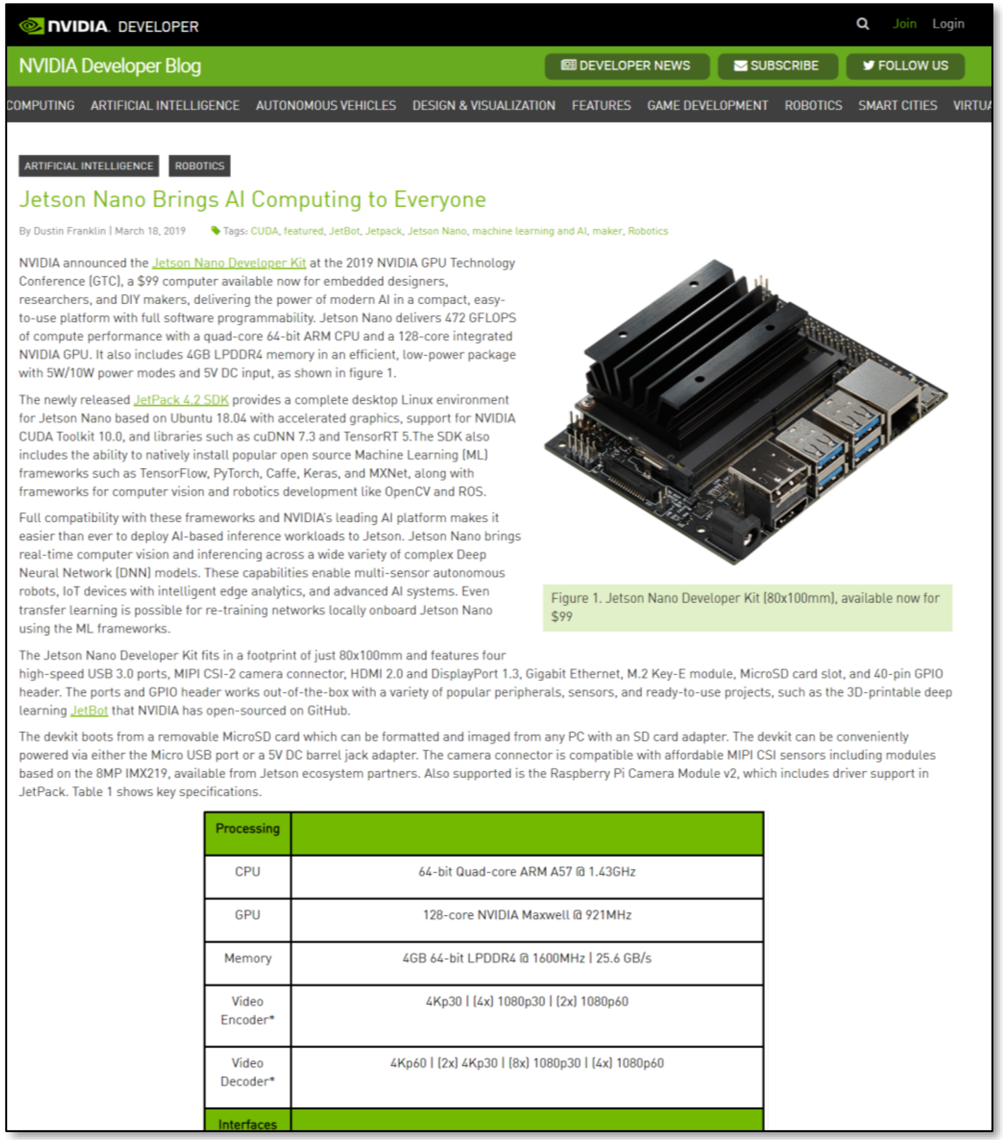
docs.nvidia.com/cuda/cuda-for-tegra-appnote/

Thank you!



Developer Site	developer.nvidia.com/jetson
Getting Started	nvidia.com/JetsonNano-Start
Hello AI World	github.com/dusty-nv
DevTalk Forums	devtalk.nvidia.com
Visit the Wiki	eLinux.org/Jetson_Nano

Q&A: What can I help you build?



Jetson Nano Brings AI Computing to Everyone

By Dustin Franklin | March 18, 2019 Tags: CUDA, featured, JetBot, Jetpack, Jetson Nano, machine learning and AI, maker, Robotics

NVIDIA announced the Jetson Nano Developer Kit at the 2019 NVIDIA GPU Technology Conference (GTC), a $99 computer available now for embedded designers, researchers, and DIY makers, delivering the power of modern AI in a compact, easy-to-use platform with full software programmability. Jetson Nano delivers 472 GFLOPS of compute performance with a quad-core 64-bit ARM CPU and a 128-core integrated NVIDIA GPU. It also includes 4GB LPDDR4 memory in an efficient, low-power package with 5W/10W power modes and 5V DC input, as shown in figure 1.

The newly released JetPack 4.2 SDK provides a complete desktop Linux environment for Jetson Nano based on Ubuntu 18.04 with accelerated graphics, support for NVIDIA CUDA Toolkit 10.0, and libraries such as cuDNN 7.3 and TensorRT 5.The SDK also includes the ability to natively install popular open source Machine Learning (ML) frameworks such as TensorFlow, PyTorch, Caffe, Keras, and MXNet, along with frameworks for computer vision and robotics development like OpenCV and ROS.

Full compatibility with these frameworks and NVIDIA's leading AI platform makes it easier than ever to deploy AI-based inference workloads to Jetson. Jetson Nano brings real-time computer vision and inferencing across a wide variety of complex Deep Neural Network (DNN) models. These capabilities enable multi-sensor autonomous robots, IoT devices with intelligent edge analytics, and advanced AI systems. Even transfer learning is possible for re-training networks locally onboard Jetson Nano using the ML frameworks.

Figure 1. Jetson Nano Developer Kit (80x100mm), available now for $99

The Jetson Nano Developer Kit fits in a footprint of just 80x100mm and features four high-speed USB 3.0 ports, MIPI CSI-2 camera connector, HDMI 2.0 and DisplayPort 1.3, Gigabit Ethernet, M.2 Key-E module, MicroSD card slot, and 40-pin GPIO header. The ports and GPIO header works out-of-the-box with a variety of popular peripherals, sensors, and ready-to-use projects, such as the 3D-printable deep learning JetBot that NVIDIA has open-sourced on GitHub.

The devkit boots from a removable MicroSD card which can be formatted and imaged from any PC with an SD card adapter. The devkit can be conveniently powered via either the Micro USB port or a 5V DC barrel jack adapter. The camera connector is compatible with affordable MIPI CSI sensors including modules based on the 8MP IMX219, available from Jetson ecosystem partners. Also supported is the Raspberry Pi Camera Module v2, which includes driver support in JetPack. Table 1 shows key specifications.

Processing	
CPU	64-bit Quad-core ARM A57 @ 1.43GHz
GPU	128-core NVIDIA Maxwell @ 921MHz
Memory	4GB 64-bit LPDDR4 @ 1600MHz | 25.6 GB/s
Video Encoder*	4Kp30 | (4x) 1080p30 | (2x) 1080p60
Video Decoder*	4Kp60 | (2x) 4Kp30 | (8x) 1080p30 | (4x) 1080p60
Interfaces	

Dev Blog *Jetson Nano Brings AI Computing to Everyone*