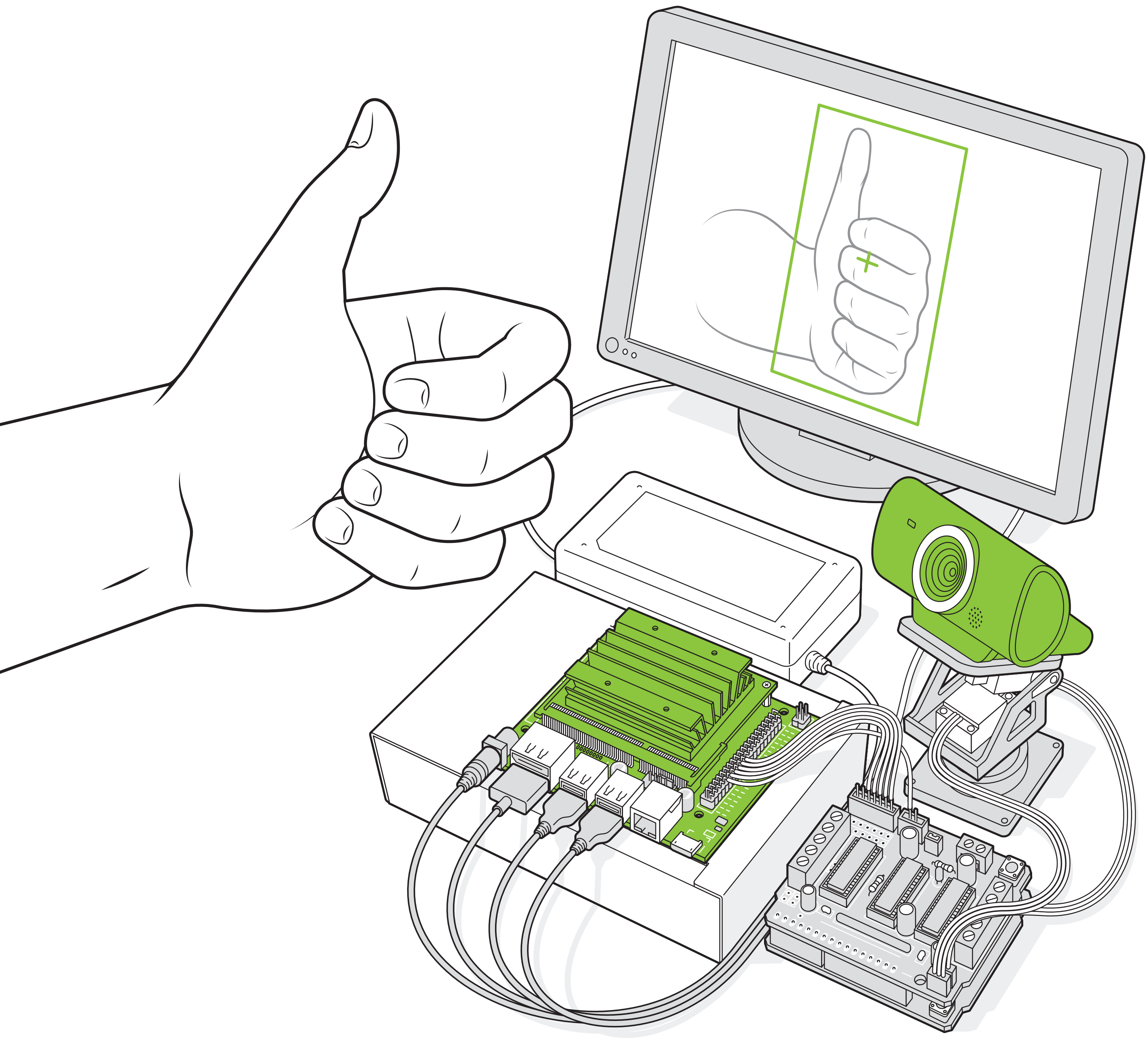
Hands On

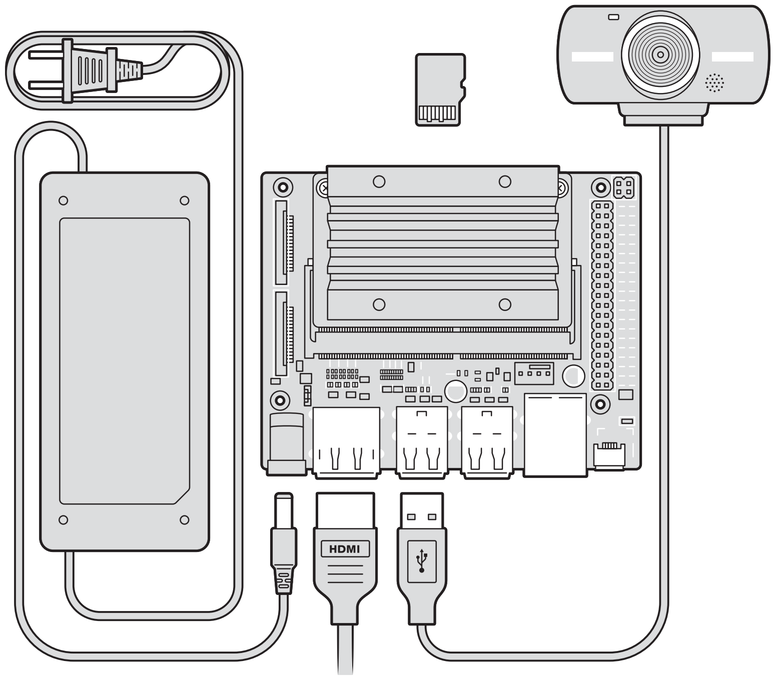


HANDS ON BY STEPHEN CASS

ILLUSTRATIONS BY James Provost

NVIDIA MAKES IT EASY TO EMBED AI

THE JETSON NANO PACKS A LOT OF MACHINE-LEARNING POWER INTO DIY PROJECTS



AI ENGINE: The Jetson Nano is the cheapest of a number of AI development kits made by Nvidia. Like many AI development boards it has hefty power requirements, so users are encouraged to buy an external power adapter that can supply at least 4 amperes. Lots of power in necessarily means lots of heat out, so the kit is dominated by a large heat sink which has screw holes for adding a fan if desired. The Nano has lots of input/output options for displays, USB peripherals, camera modules, and a Raspberry Pi–compatible 40-pin general purpose input/output header.

When opportunity knocks, open the door: No one has taken heed of the adage like Nvidia, which has transformed itself from a company focused on catering to the needs of video gamers to one at the heart of the artificial-intelligence revolution. In 2001, no one predicted that the same processor architecture developed to draw realistic explosions in 3D would be just the thing to power a renaissance in deep learning. But when Nvidia realized that academics were gobbling up its graphics cards, it responded, supporting researchers with the launch of the CUDA parallel computing software framework in 2006.

Since then, Nvidia has been a big player in the world of high-end embedded AI applications, where teams of highly trained (and paid) engineers have used its hardware for things like autonomous vehicles. Now the company claims to be making it easy for even hobbyists to use embedded machine learning, with its US $100 Jetson Nano dev kit, which was originally launched in early 2019 and rereleased this March with several upgrades. So, I set out to see just how easy it was: Could I, for example, quickly and cheaply make a camera that could recognize and track chosen objects?

Embedded machine learning is evolving rapidly. In April 2019, Hands On looked at Google's Coral Dev AI board which incorporates the company's Edge tensor processing unit (TPU), and in July 2019, *IEEE Spectrum* featured Adafruit's software library, which lets even a handheld game device do simple speech recognition. The Jetson Nano is closer to the Coral Dev board: With its 128 parallel processing cores, like the Coral, it's powerful enough to handle a real-time video feed, and both have Raspberry Pi–style 40-pin GPIO connectors for driving external hardware.

But the Coral Dev board is designed to provide the minimal amount of support needed to prototype embedded applications. Its Edge TPU is on a removable module that can be plugged directly into target hardware as development progresses. The Coral carrier board that the processor module sits on has only one large USB port and is generally intended to be operated over a remote connection via a USB-C or Ethernet connector rather than being plugged in to a keyboard and screen directly.

The Jetson Nano has a similar module-and-carrier board approach, but its carrier board has a lot more going for stand-alone experimentation. It has four USB 3.0 ports for peripherals, HDMI and DisplayPort connectors, a Micro-USB port to supply power or allow remote operation, an Ethernet port, two ribbon connectors for attaching Raspberry Pi–compatible camera modules, and a barrel jack socket for providing the additional power needed for intensive computations. (I appreciated this last touch, because the Coral Dev board uses two USB-C ports side by side, one for power and the other for data, and I was always confusing the two.)

Nvidia's "Getting Started" instructions stepped me through downloading a customized version of the Ubuntu operating system designed for the Nano. This includes things like a Nano version of the established Raspberry Pi Rpi.GPIO library for accessing the GPIO header from Python scripts.

Then, as per the instructions, I began the free "Welcome to Getting Started With AI on Jetson Nano!" course. This includes an introductory primer on neural networks, followed by details on how to use the Nano in remote mode. In this mode, the Nano runs a local Web server that hosts a Jupyter notebook, the tool of choice for many researchers because it allows explanatory text and images to be intermingled with Python code.

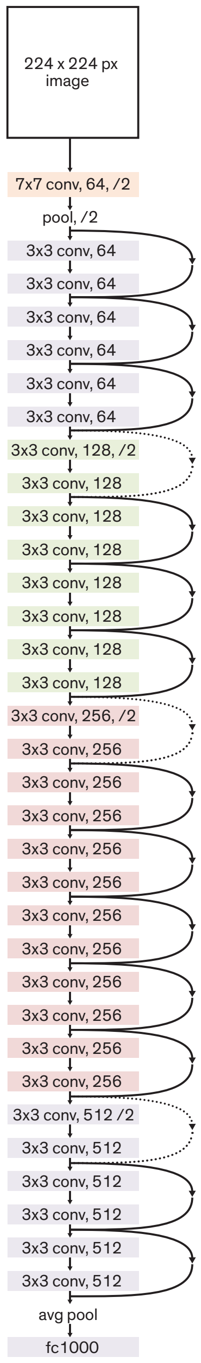
Unfortunately, I couldn't connect to the Nano's server: After some poking around online, it turns out that the Ubuntu version required for the "Welcome…" course is different from the version linked in the "Getting Started" instructions. There are actually a few places where Nvidia's generally excellent documentation, videos, and other tutorials have this kind of rough edge: Sometimes it's because things haven't been updated for the new board; other times it's simple mistakes, such as a sample script used for testing the output of the GPIO header that tries to import the original Rpi.GPIO library rather than the Jetson.GPIO library.

But soon I was up and running with the well-known ResNet-18 pretrained neural network, which can classify 1,000 objects. Through a provided notebook it is easy to modify the last layer of the network so that it recognizes just two things—say, thumbs-up and thumbs-down gestures—using images directly captured from the USB camera, and then retrain the network for gesture recognition.

Then I tried something a little more ambitious. Following along with Nvidia's suggested "Hello AI World" tutorial, I was soon running a 10-line Python script directly on the Nano's desktop that did real-time recognition with the ResNET-18 network. With a little bit of tweaking, I was able to extract the ID and center coordinates of every object recognized. A little more modification and the script toggled four GPIO pins to indicate if a particular type of object—say a bottle, or person—was above, below, left of, or right of center in the camera's view.

I hooked up an Arduino Uno with a motor shield to drive the two servos in a $19 pan/tilt head that I bought from Adafruit. I mounted my USB camera on the pan/tilt head and wired the Arduino to monitor the Nano's GPIO pins. When the Nano senses an off-center object and turns on the appropriate GPIO pin, the Arduino nudges the pan/tilt head in the direction that should bring the object toward the center of its field of view.

I was genuinely surprised how quickly I was able to go from opening the Nano's box to a working autonomous tracking camera (not counting the time I spent waiting for various supporting bits and pieces to arrive in the mail while isolated from my usual workbench at the *Spectrum* office!). And there are plenty of avenues for further expansion and experimentation. If you're looking to merge physical computing with AI, the Jetson Nano is a terrific place to start. **—STEPHEN CASS**



MULTILAYERED ANALYSIS: The software suite provided by Nvidia makes it easy to download many popular pretrained neural-network architectures that are suited for different tasks, such as identifying pedestrians or household objects. The ResNet-18 network pictured above can identify 1,000 such objects. You can make the network identify new, similar, objects by reconfiguring the final classification layer, which attaches labels to objects, and partially retraining the network.

POST YOUR COMMENTS AT
spectrum.ieee.org/jetson-jul2020