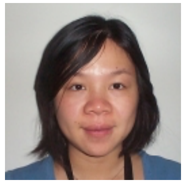
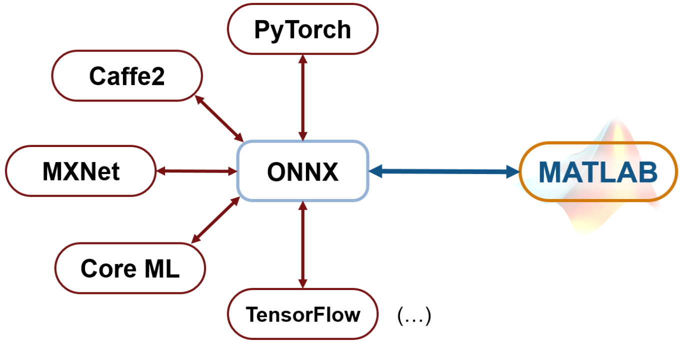
ONNX with MATLAB

Shounak Mitra
Product Manager
Deep Learning



Ting Su
Development Manager
Deep Learning



Open Neural Network Exchange

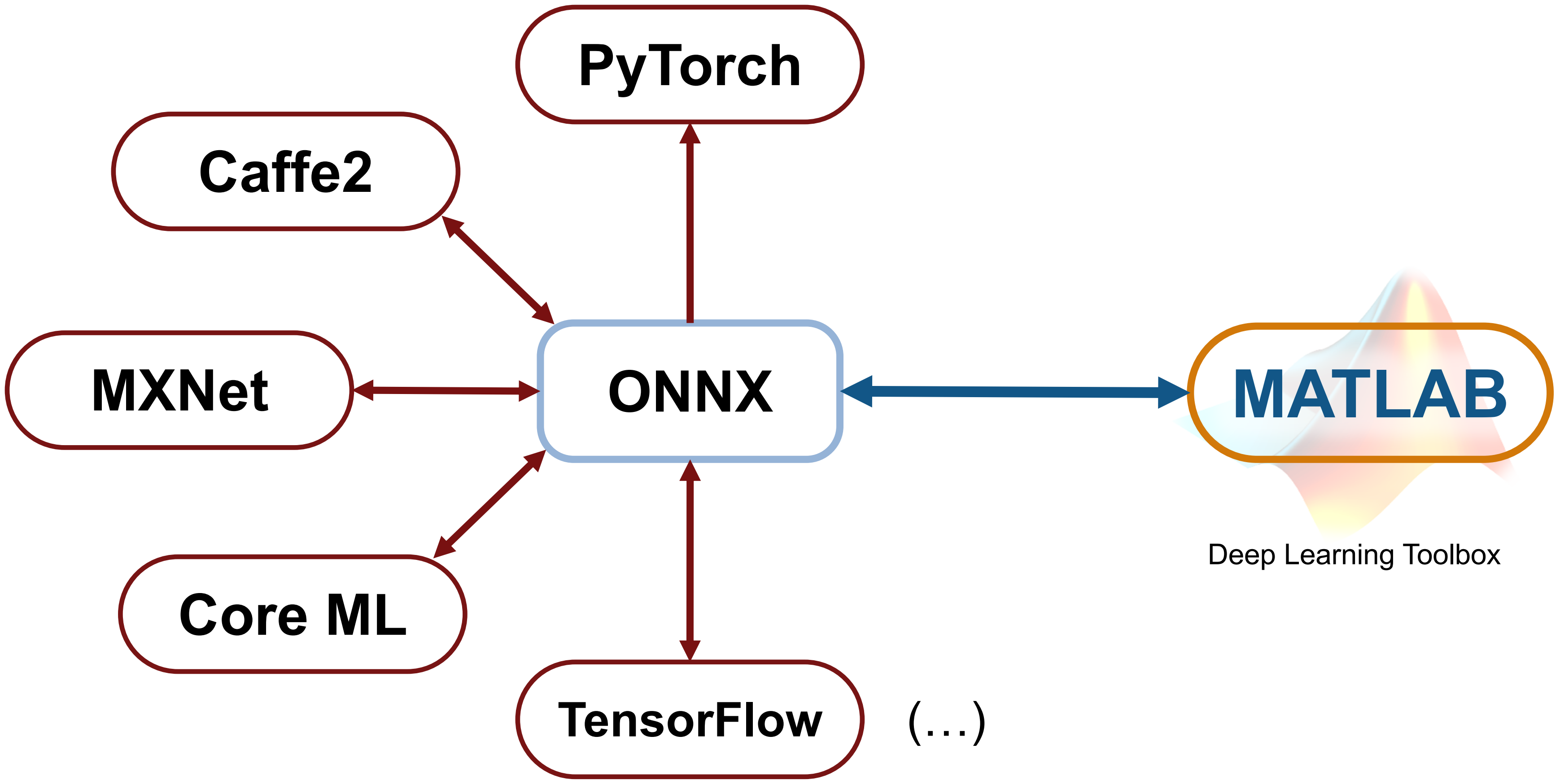
Agenda

- MathWorks’ Investments in ONNX
- How MATLAB users use ONNX
- Current Goals

Agenda

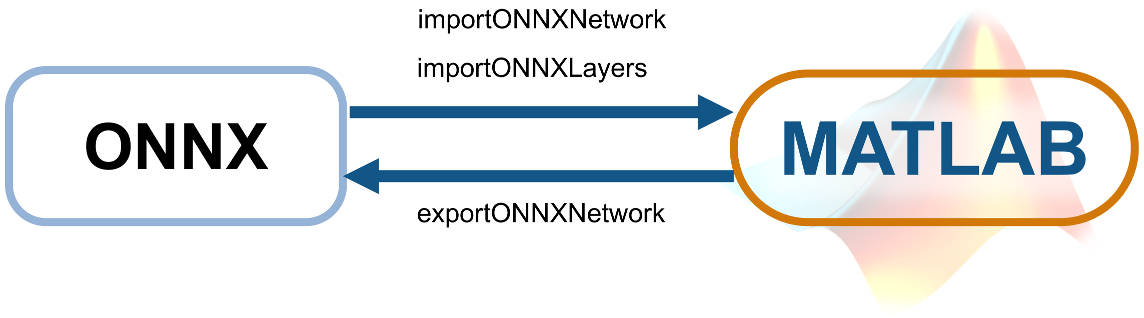
- MathWorks’ Investments in ONNX
- How MATLAB users use ONNX
- Current Goals

Model exchange with MATLAB



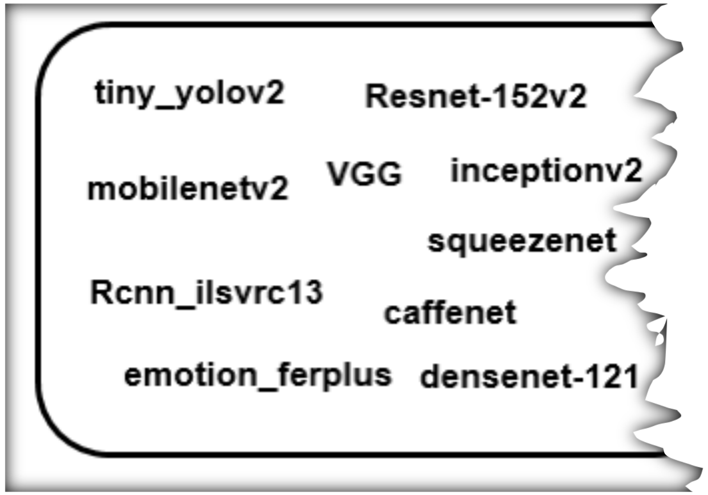
Open Neural Network Exchange

MATLAB’s support for ONNX



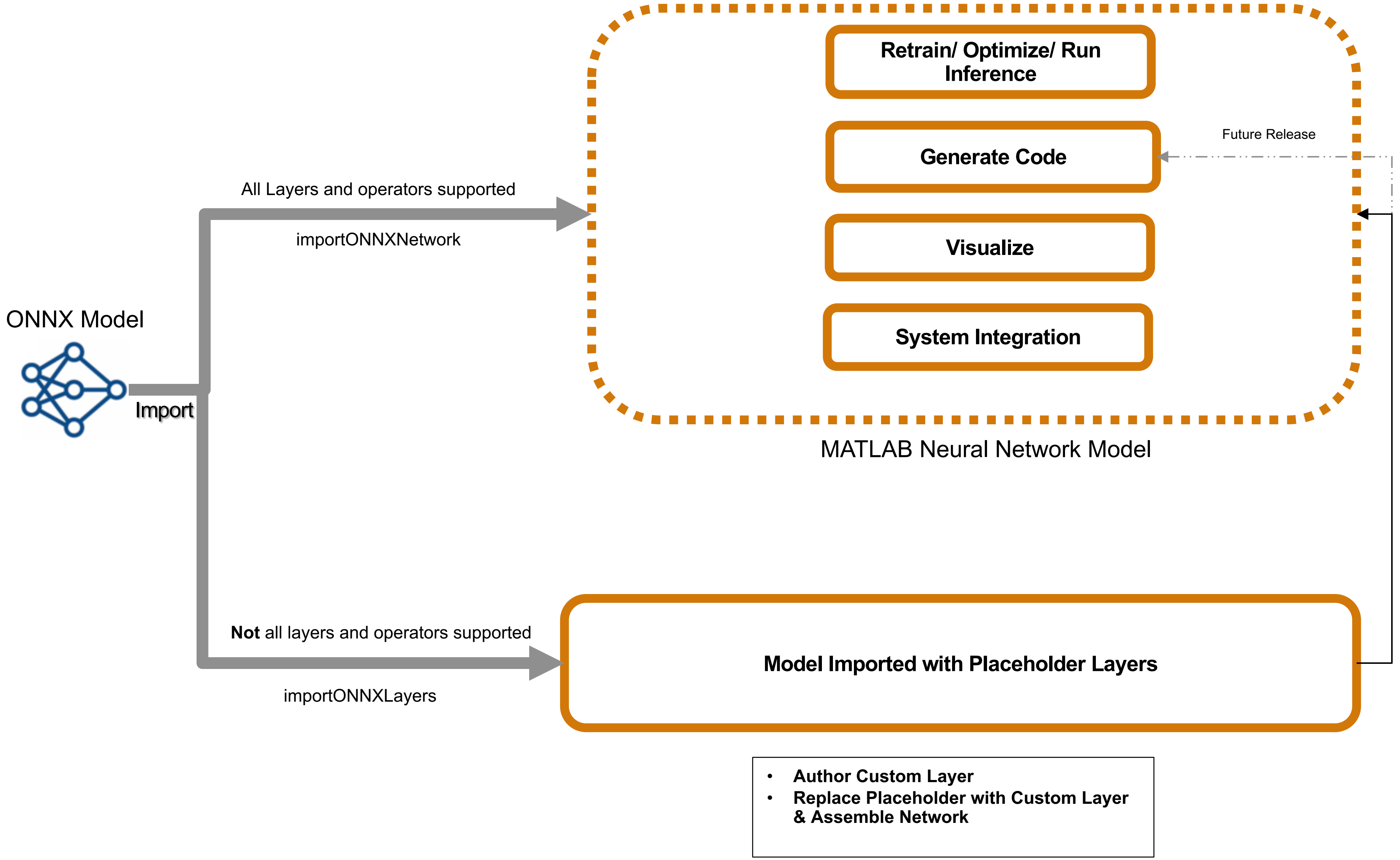
ONNX Layer
Add
AveragePool
BatchNormalization
Clip
Concat
Constant
Conv
ConvTranspose
Div
Dropout

Increasing number of support for ONNX layers



Support for Models from ONNX Model Zoo

ONNX Model Import Workflow in MATLAB



Agenda

- MathWorks' Investments in ONNX
- How MATLAB users use ONNX
- Current Goals

Export to ONNX Workflow: Users use MATLAB for Labeling, Preprocessing, Training

(Automatic) Labeling → **Preprocessing** → **Training and Experiment Management** → exportONNXNetwork → **ONNX Model Format**

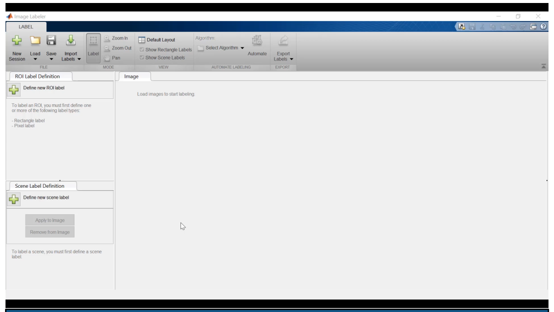
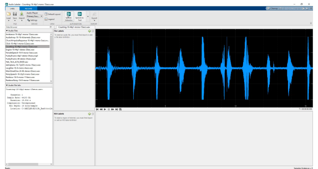
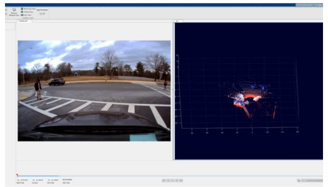
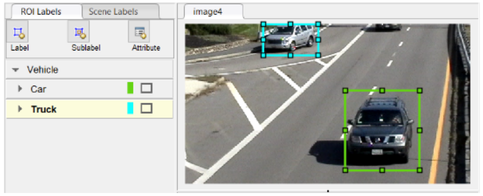


Image Labeler App: Automate Image Labeling


Signal Labeler

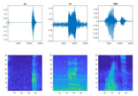

LiDar Point Cloud Labeling



Video Labeling


Computer Vision


Control Design


Audio Processing

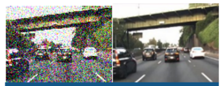
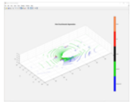
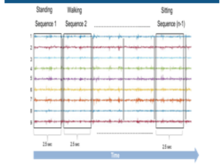


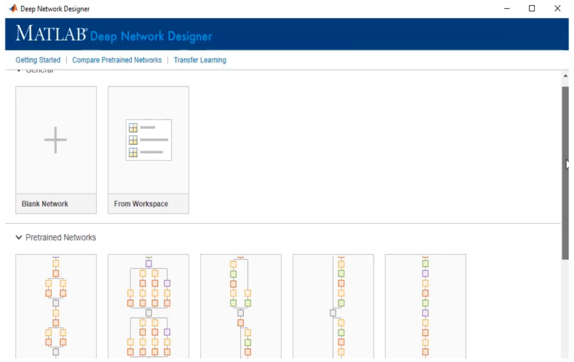
Image Processing


Lidar Processing

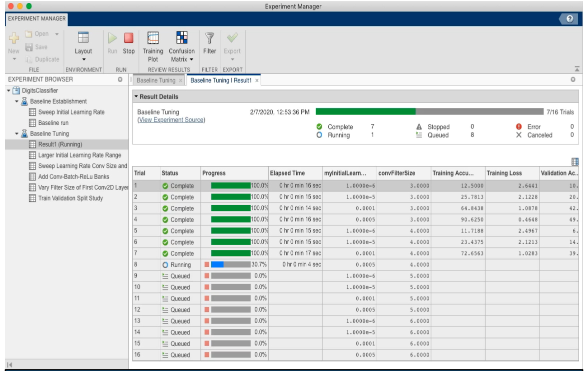

N-D Volumes


Text Analytics


Sensor Data Analysis



Deep Network Designer App: Point and Click tool to design and train networks


Experiment Manager App

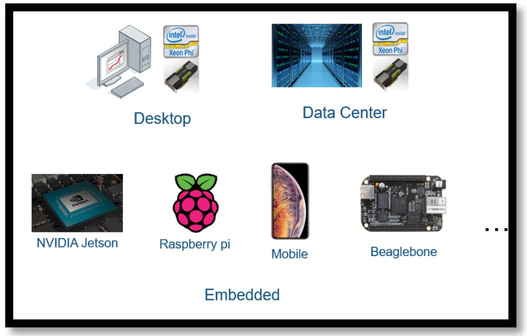
Import from ONNX Workflow: Users use MATLAB for Code Generation, Visualization, Re-Training, and Simulink for System Integration

ONNX Model

importONNXNetwork

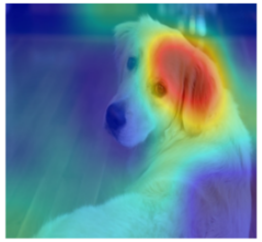
MATLAB Neural Network Model

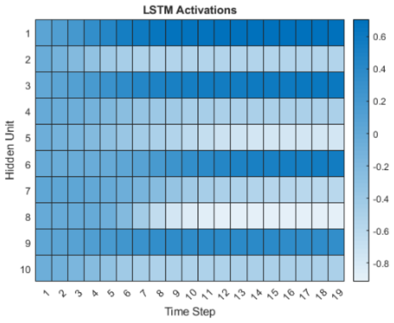
Code Generation and Deployment



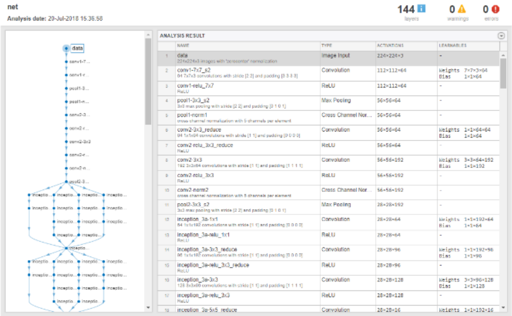
Visualization/ Debugging

Grad-CAM



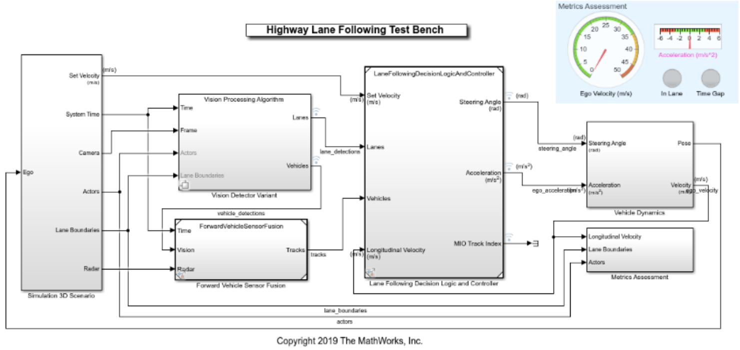


Analyze Network/ Retrain



- Automatic Differentiation
- Custom Training Loop
- Weight Sharing

System Integration (with Simulink)



- Reinforcement Learning
- Automated Driving
- Control Systems

Denso Ten Develops Model-Based Workflow Process for AI Control System Development

Natsuki Yokoyama, Denso Ten

Complex vehicle control issues can be difficult to formulate and require the experience of experts to resolve. Deep learning is often used to overcome this challenge. Denso Ten initially used Python for their AI work, but they had no way to convert it to C code for ECU implementation and couldn't apply it to simulation for model-based development. Instead, they developed a model-based workflow using MATLAB® and Simulink®.

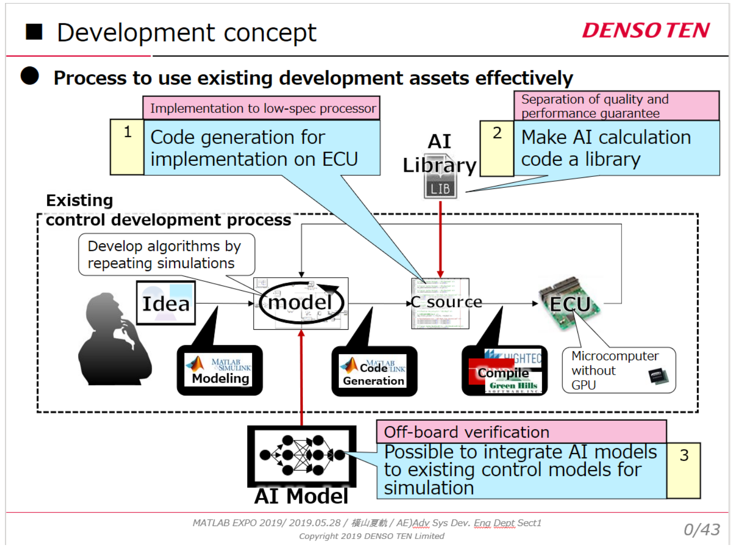
They begin by importing the AI model created in MATLAB® into the existing Simulink® control system model. Using the Simulink API, they can automatically create Simulink blocks, connect wires, and copy weight values from AI models. Denso Ten also developed an AI library and can now simulate the entire model. They then convert the Simulink model back to a MATLAB based AI model.

Denso Ten has completed the model-based development workflow from design to implementation; they are now proceeding with development for production.

Advantages of using MATLAB and Simulink:

- Apply model-based development to integrate AI model into existing control model
- Use Deep Network Designer for network construction through mouse operations
- Use API for bidirectional conversion of deep learning model between MATLAB and Simulink
- Access original AI library using S-functions

> "A model-based development workflow is essential in order to use AI for control ECUs. Combining the existing control model and the AI model enables us to establish a simulation environment and accelerate product development."



[Explore MATLAB for deep learning](#)

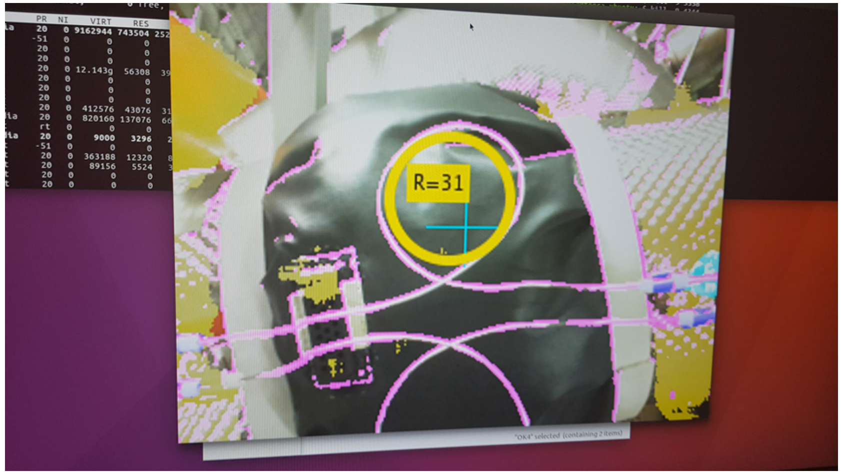
Artificial Intelligence & Deep Learning for Automatic Defect Detection

Nicolas Castet, Airbus

How do you build a robust end-to-end AI model to automatically detect the multiple defects of the pipes? That was the big challenge for Nicolas and Airbus, who used MATLAB and its solutions to quickly prototype and develop deep learning models to meet their needs.

Working with the MathWorks Consulting team, they adopted MATLAB to address the three main steps in the process. The first step was to have an integrated tool to build and train deep learning models from scratch, for approaches such as semantic segmentation, as well as an easy and interactive environment for labeling videos. With MATLAB, from position of holes on pipe, they measured distance and angle required by standard. Next, they needed to be able to display in real-time the analysis of the defects. The final step was to translate the MATLAB code to CUDA code automatically, without requiring any coding skills, to deploy it directly on the embedded system.

> "Having the possibility to **test**, **modify**, **train** and **test again** the code in a **short timeframe** was key to success."



Advantages of using MATLAB:

- An integrated tool to design, train and deploy deep learning models
- Interactive prototyping and testing in a very short amount of time
- Direct translation from MATLAB language to CUDA code

» **Learn about deep learning modelling**

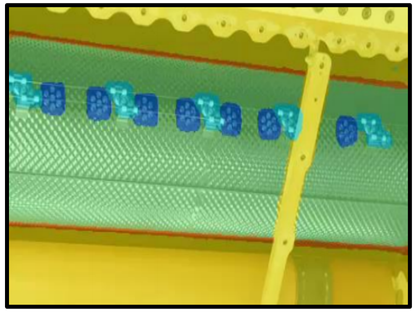
Presented at MATLAB Expo 2019 France

Airbus used MATLAB toolchain for Automatic Defect Detection of location of structural elements to calculate acceptable flex of wing of A380

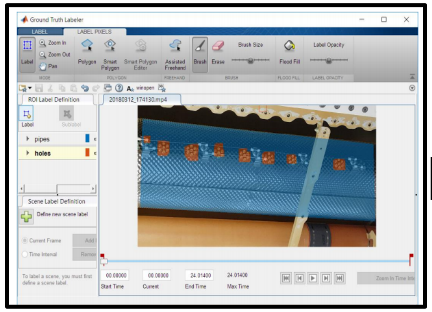
Problem Statement: Calculating acceptable flex of wing of A380.

How it worked

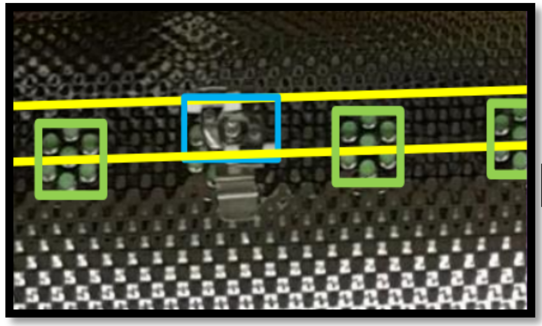
MATLAB 2019 expo talk



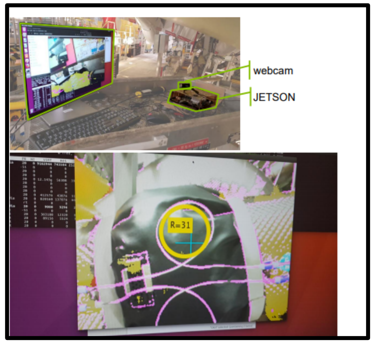
Data Visualization



Video Labeling



Model Development



GPU Deployment

Agenda

- MathWorks' Investments in ONNX
- How MATLAB users use ONNX
- Current Goals

Goals - MATLAB's Support for ONNX

- Import 90% of the models released on ONNX model zoo
- Support quantization and multi-platform code generation for imported ONNX models
- Export 90% of deep learning models trained in MATLAB to ONNX