

MATLAB EXPO 2018

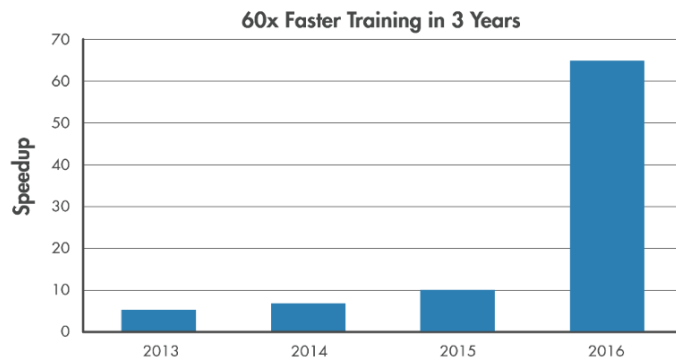
Deploying Deep Learning Networks to Embedded GPUs and CPUs

Pierre Nowodzienski

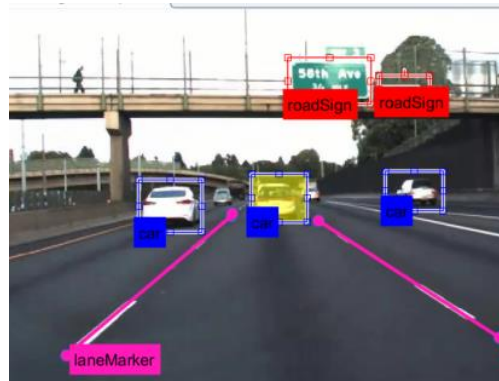
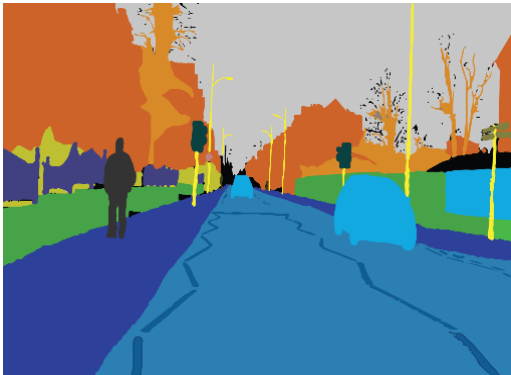


Deep Learning enablers

Increased GPU acceleration



Labeled public datasets



World-class models to be leveraged

AlexNet
PRETRAINED MODEL

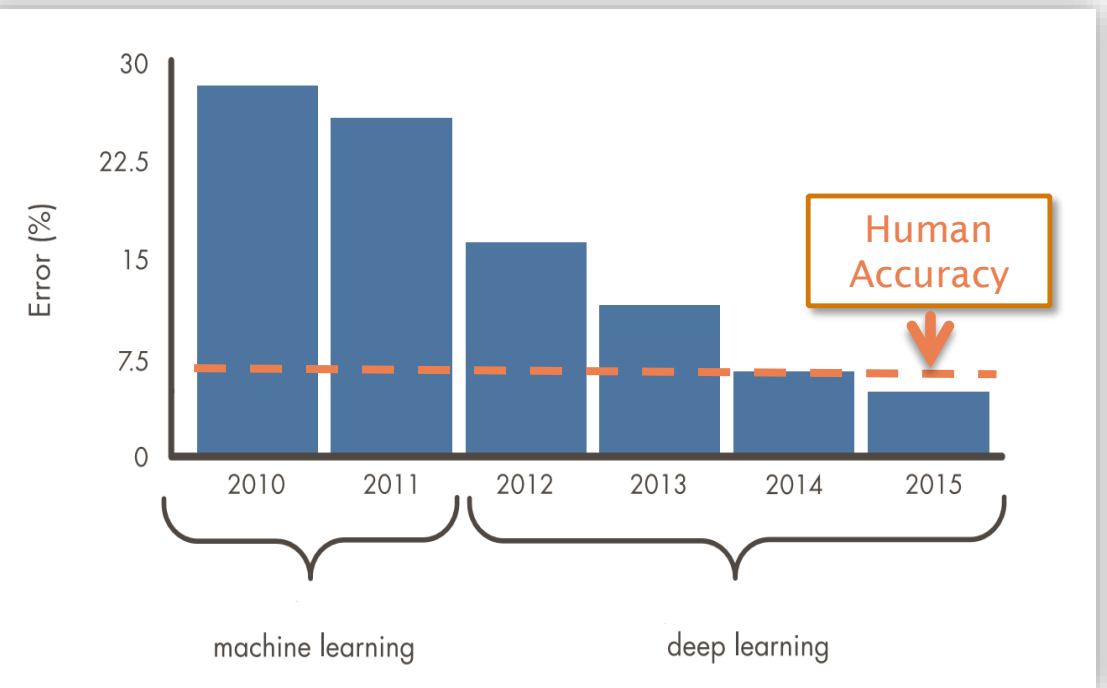
VGG-16
PRETRAINED MODEL

ResNet
PRETRAINED MODEL

Caffe
MODELS

GoogLeNet
PRETRAINED MODEL

TensorFlow/Keras
MODELS



Deep Learning Applications: Image classification, speech recognition, autonomous driving, etc...

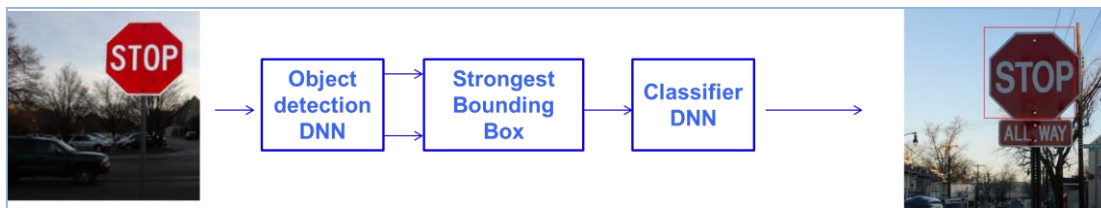


Detection of cars and road in autonomous driving systems



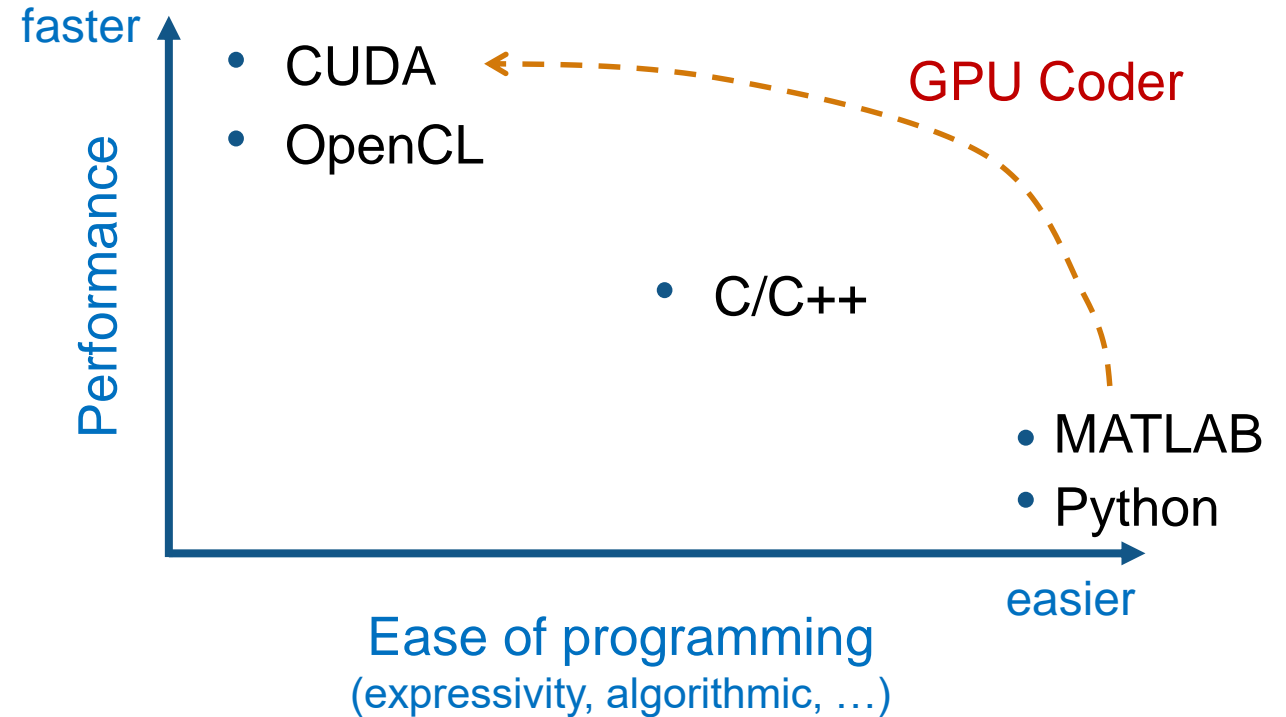
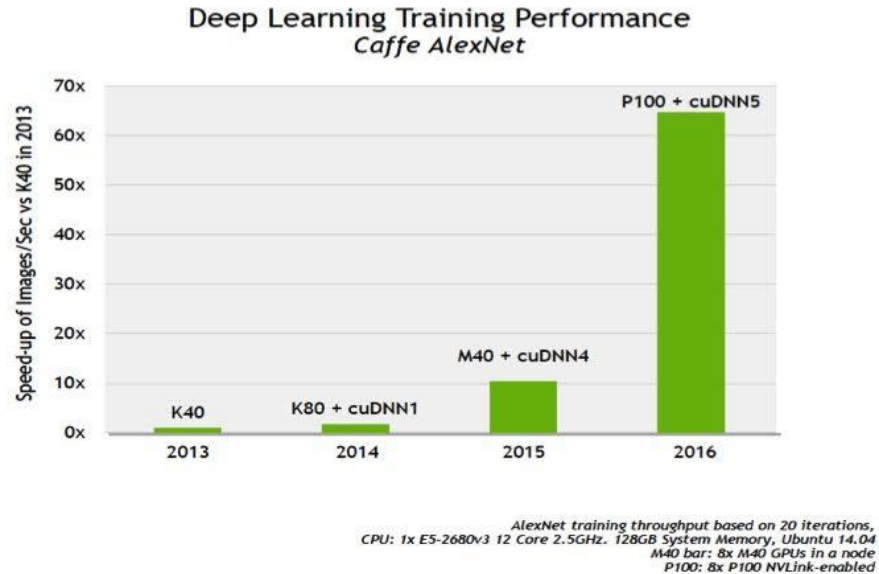
Rain Detection and Removal¹

1. *Deep Joint Rain Detection and Removal from a Single Image*



Traffic Sign Recognition

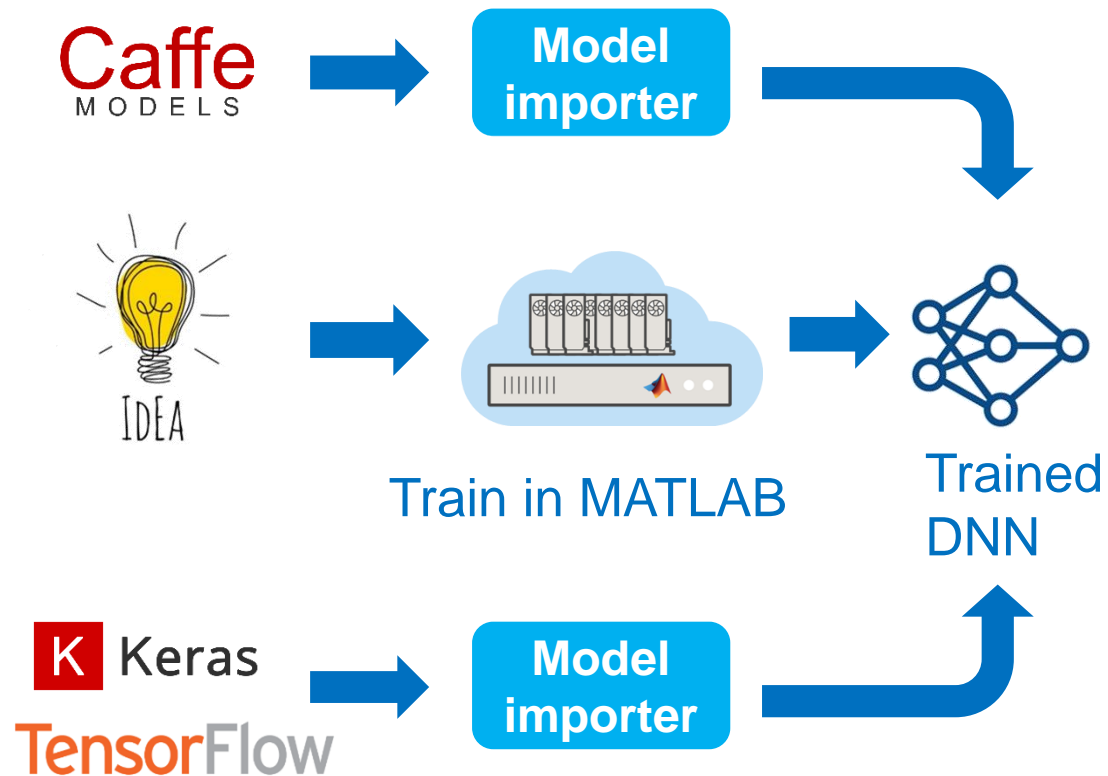
GPUs and CUDA programming



GPUs are “hardware on steroids”, ... but, programming them is hard

Deep learning workflow in MATLAB

Deep Neural Network Design + Training



■ Design in MATLAB

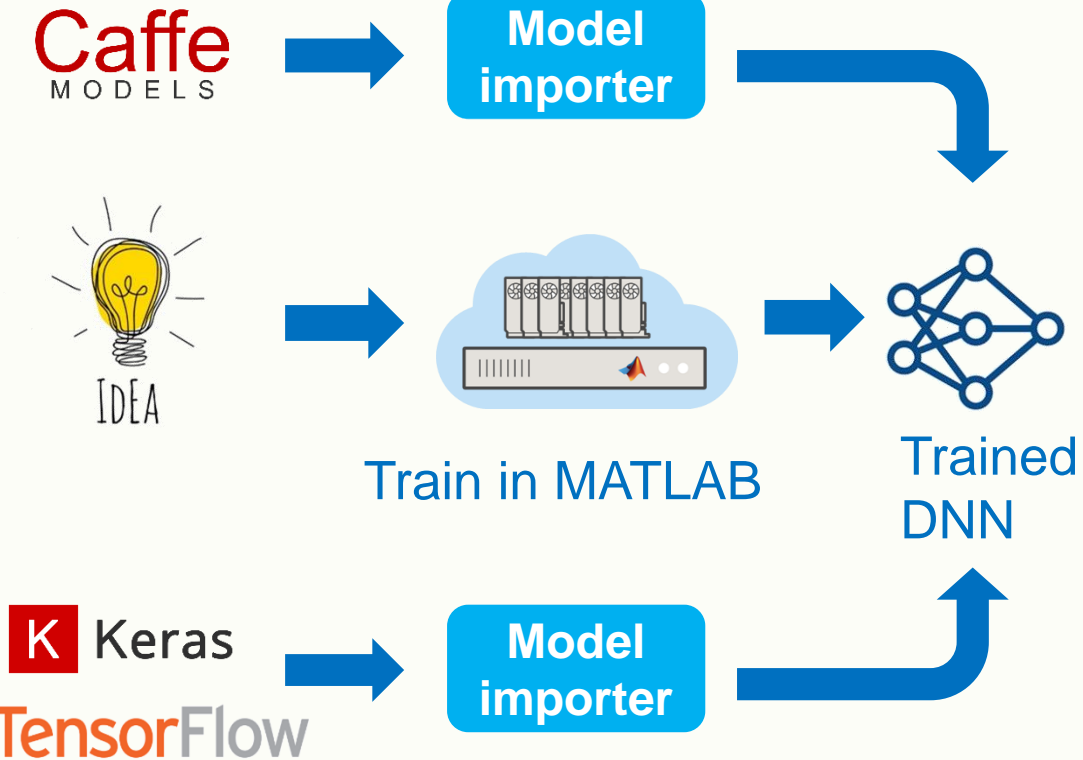
- **Manage** large data sets
- **Automate** data labeling
- **Easy access** to models

■ Training in MATLAB

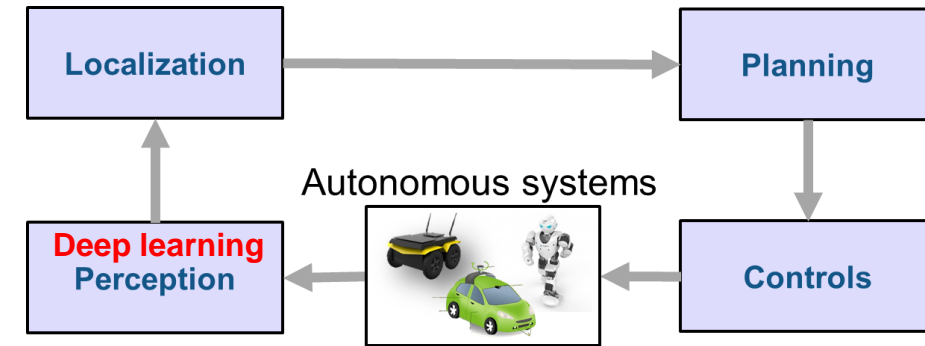
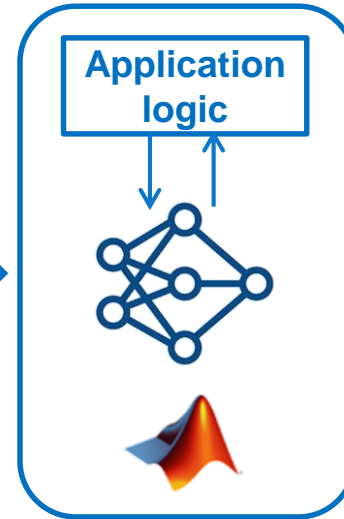
- **Acceleration** with GPU's
- **Scale** to clusters

Deep learning workflow in MATLAB

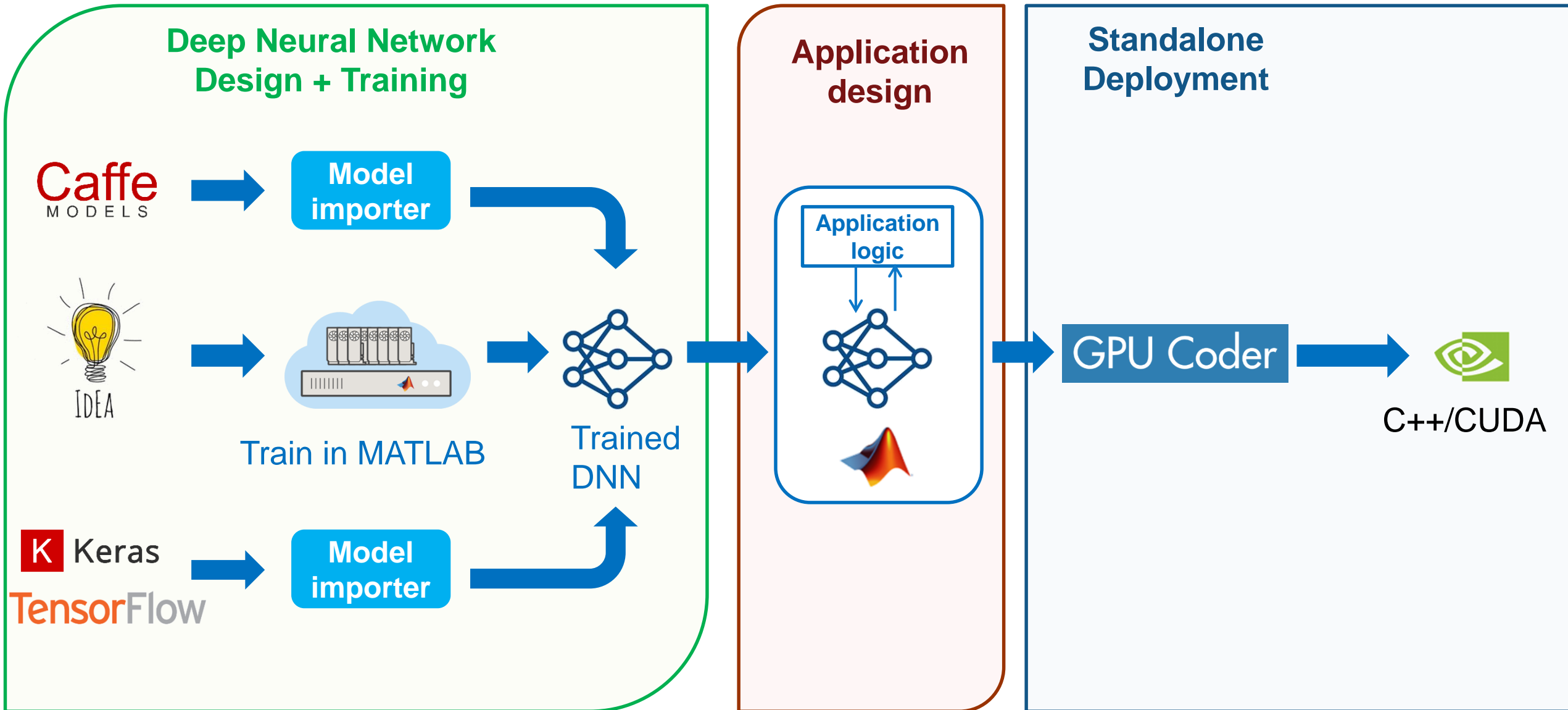
Deep Neural Network Design + Training



Application design



Deep learning workflow in MATLAB



GPU Coder for Deployment



GPU Coder



**NVIDIA
CUDA®
C/C++**



**ARM Compute
Library**

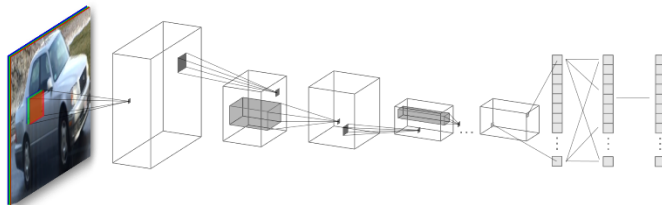
**Intel
MKL-DNN
Library**



Accelerated implementation of parallel algorithms on GPUs & CPUs

Deep Neural Networks

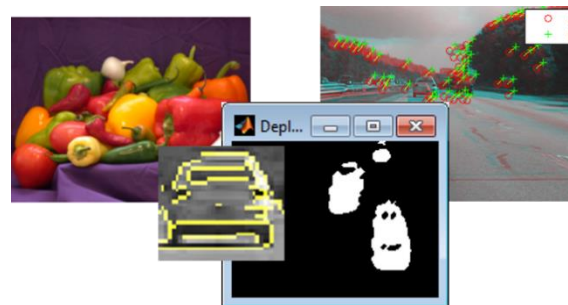
Deep Learning, machine learning



5x faster than TensorFlow
2x faster than MXNet

Image Processing and Computer Vision

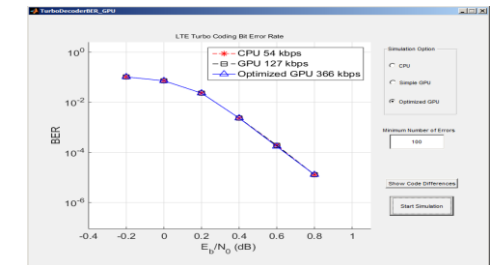
Image filtering, feature detection/extraction



60x faster than CPUs
for stereo disparity

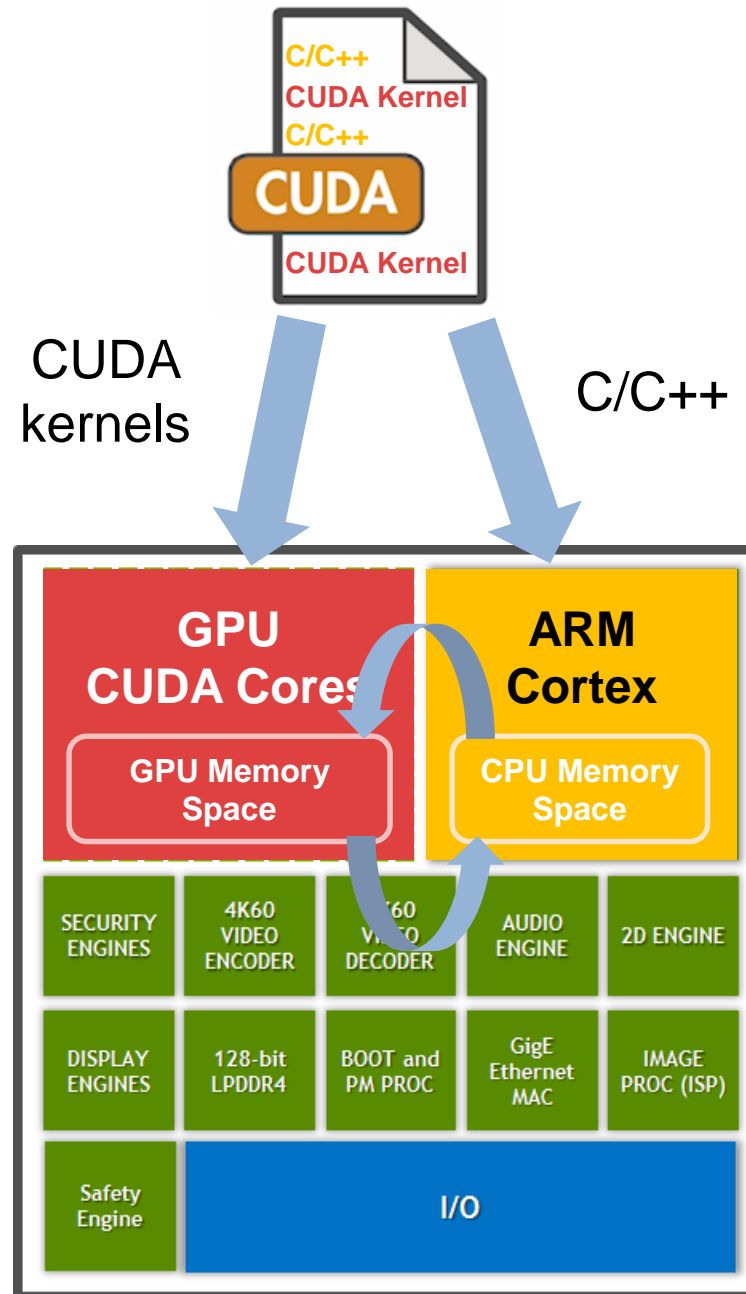
Signal Processing and Communications

FFT, filtering, cross correlation,



20x faster than CPUs
for FFTs

GPUs and CUDA



Challenges of Programming in CUDA for GPUs

- Learning to program in CUDA
 - Need to rewrite algorithms for parallel processing paradigm
- Creating CUDA kernels
 - Need to analyze algorithms to create CUDA kernels that maximize parallel processing
- Allocating memory
 - Need to deal with memory allocation on both CPU and GPU memory spaces
- Minimizing data transfers
 - Need to minimize while ensuring required data transfers are done at the appropriate parts of your algorithm

GPU Coder Helps You Deploy to GPUs Faster



GPU Coder

CUDA Kernel creation

Memory allocation

Data transfer minimization



NVIDIA.
CUDA®
C/C++

- Library function mapping
- Loop optimizations
- Dependence analysis

- Data locality analysis
- GPU memory allocation

- Data-dependence analysis
- Dynamic memcopy reduction

GPU Coder speeds up MATLAB for Image Processing and Computer Vision



Fog removal



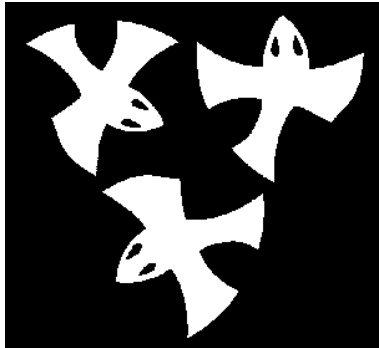
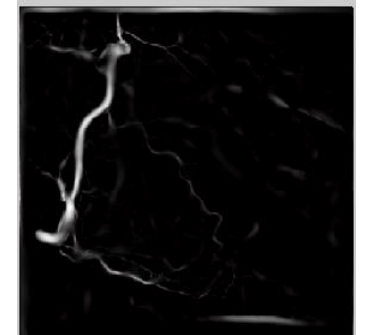
5x speedup



Frangi filter



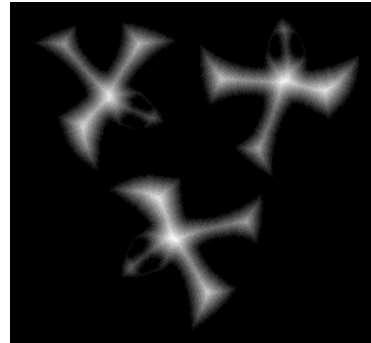
3x speedup



Distance transform



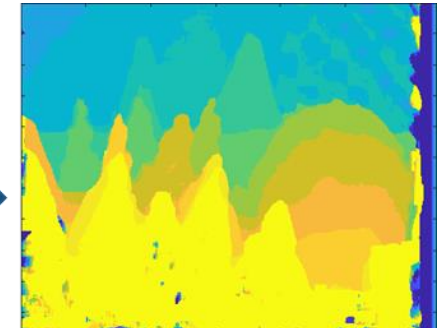
8x speedup



Stereo disparity



50x speedup



Ray tracing



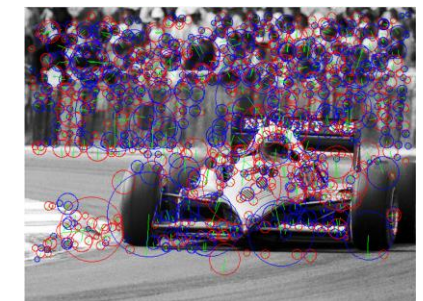
18x speedup



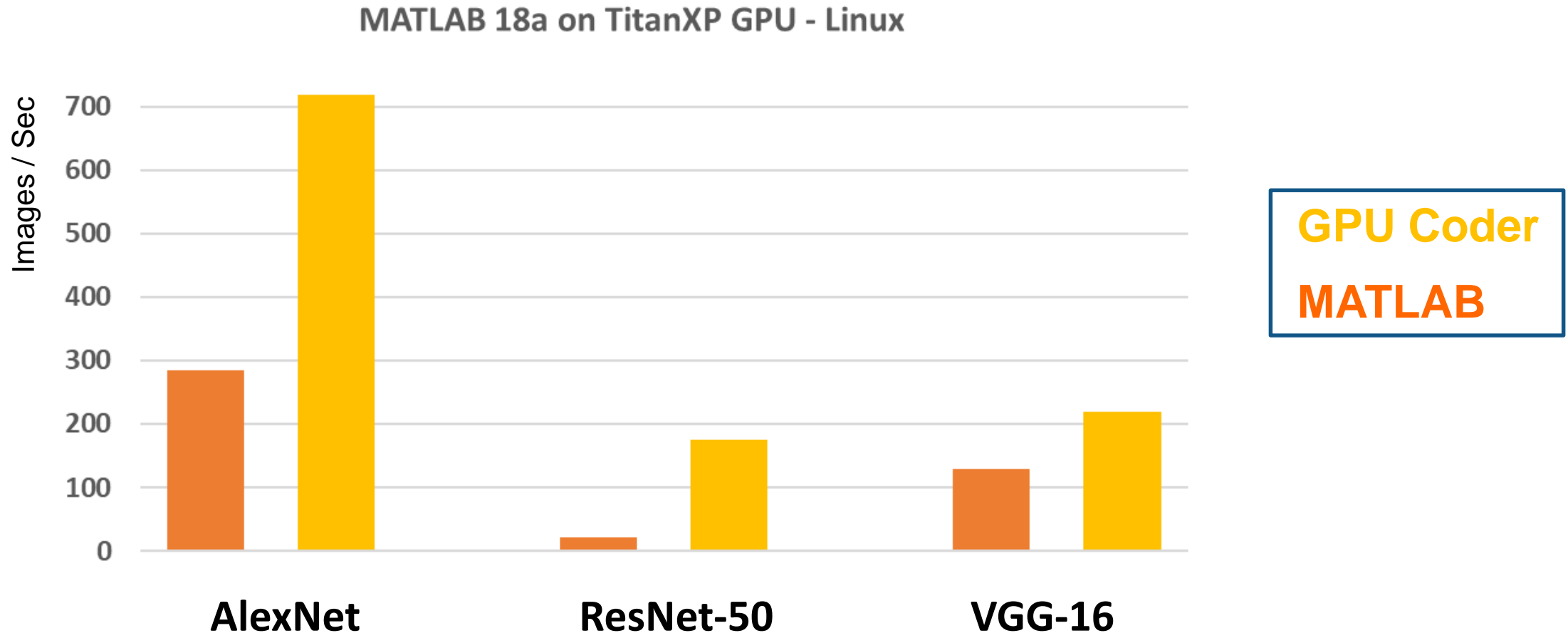
SURF feature extraction



700x speedup



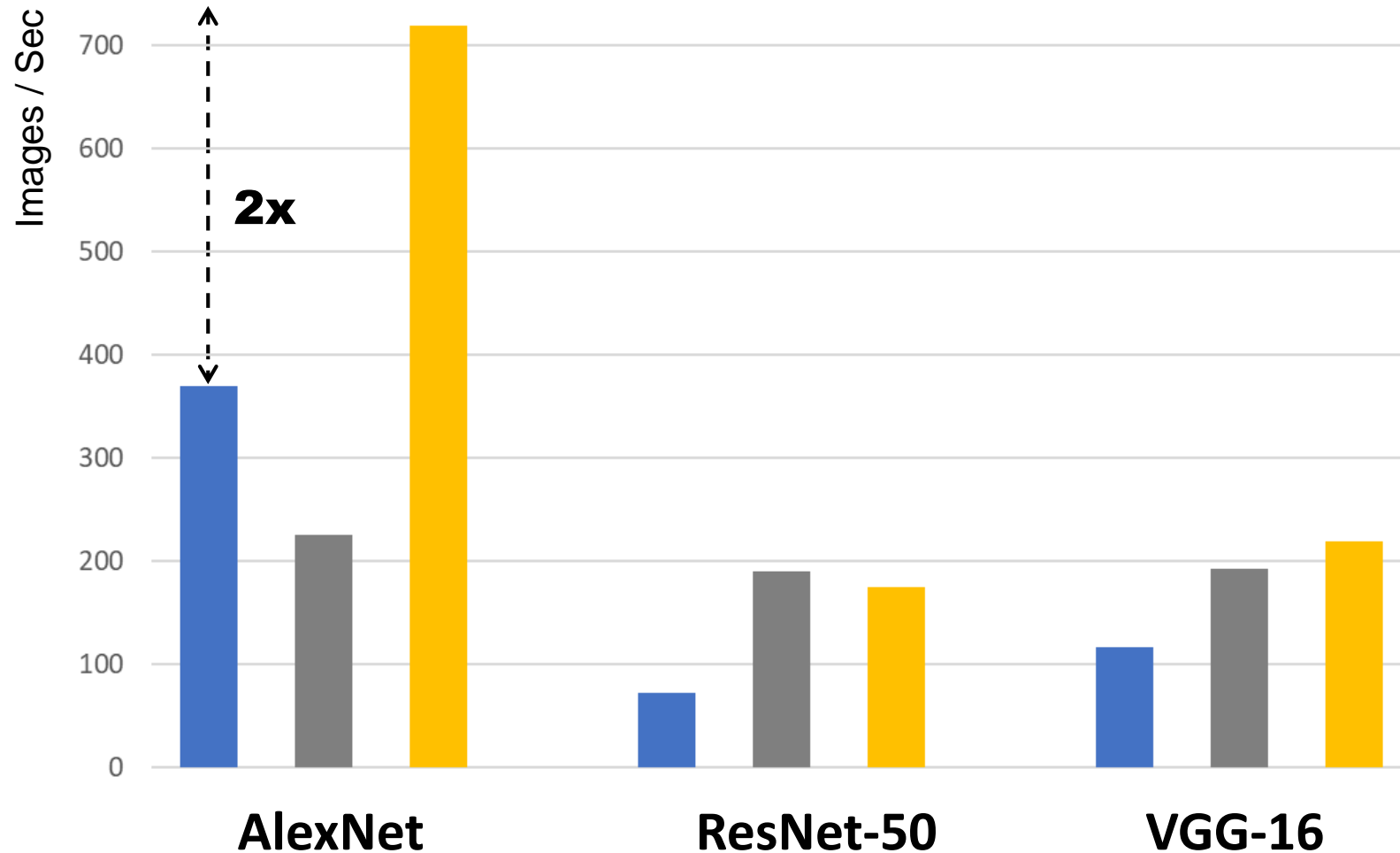
GPU Coder speeds up MATLAB at least 2x for inference



Single image prediction using Intel® Xeon® CPU - 3.6 GHz, NVIDIA libraries: CUDA8 - cuDNN 7
TensorFlow 1.6.0, MXNet 1.1.0, MATLAB 18a

With GPU Coder, MATLAB is faster than other frameworks

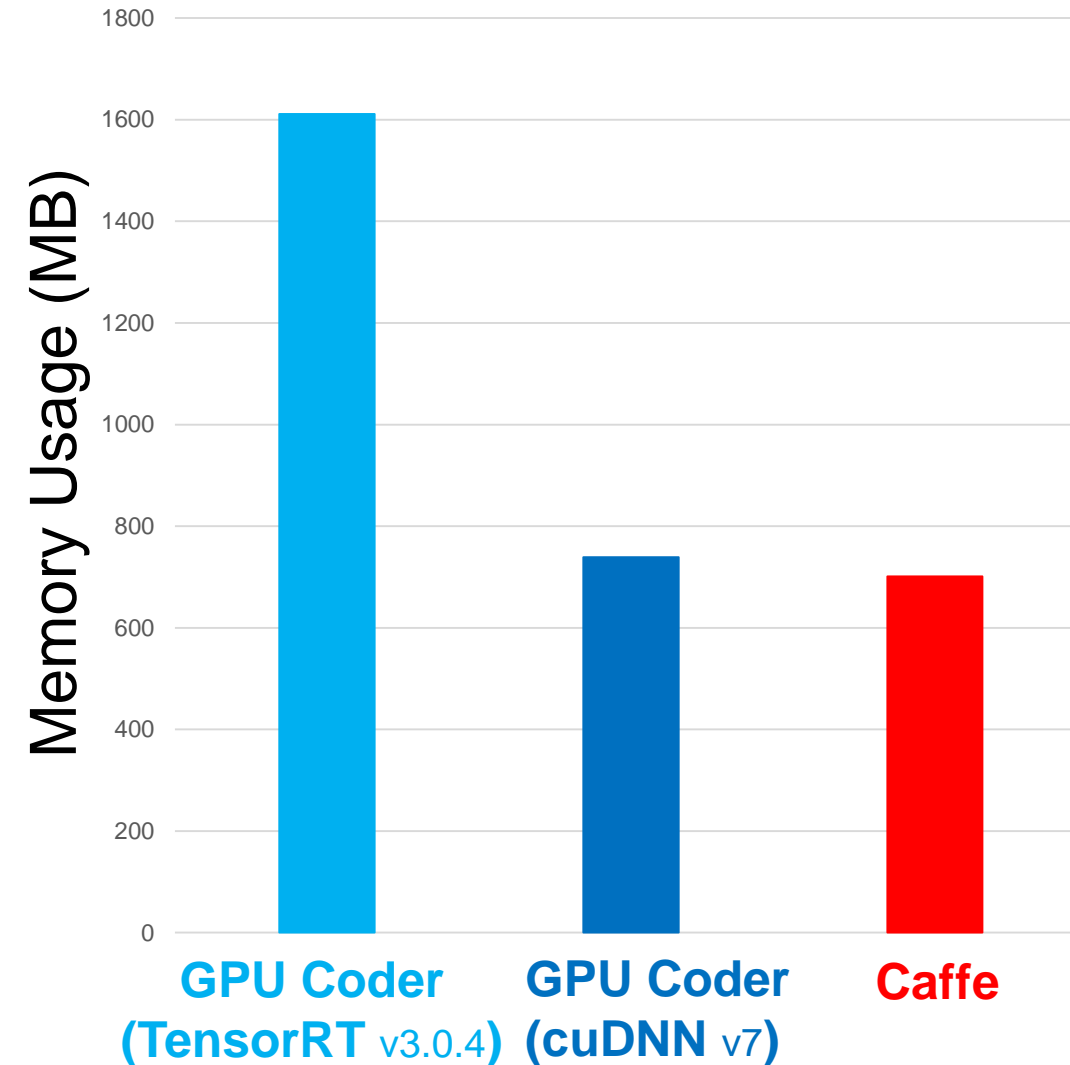
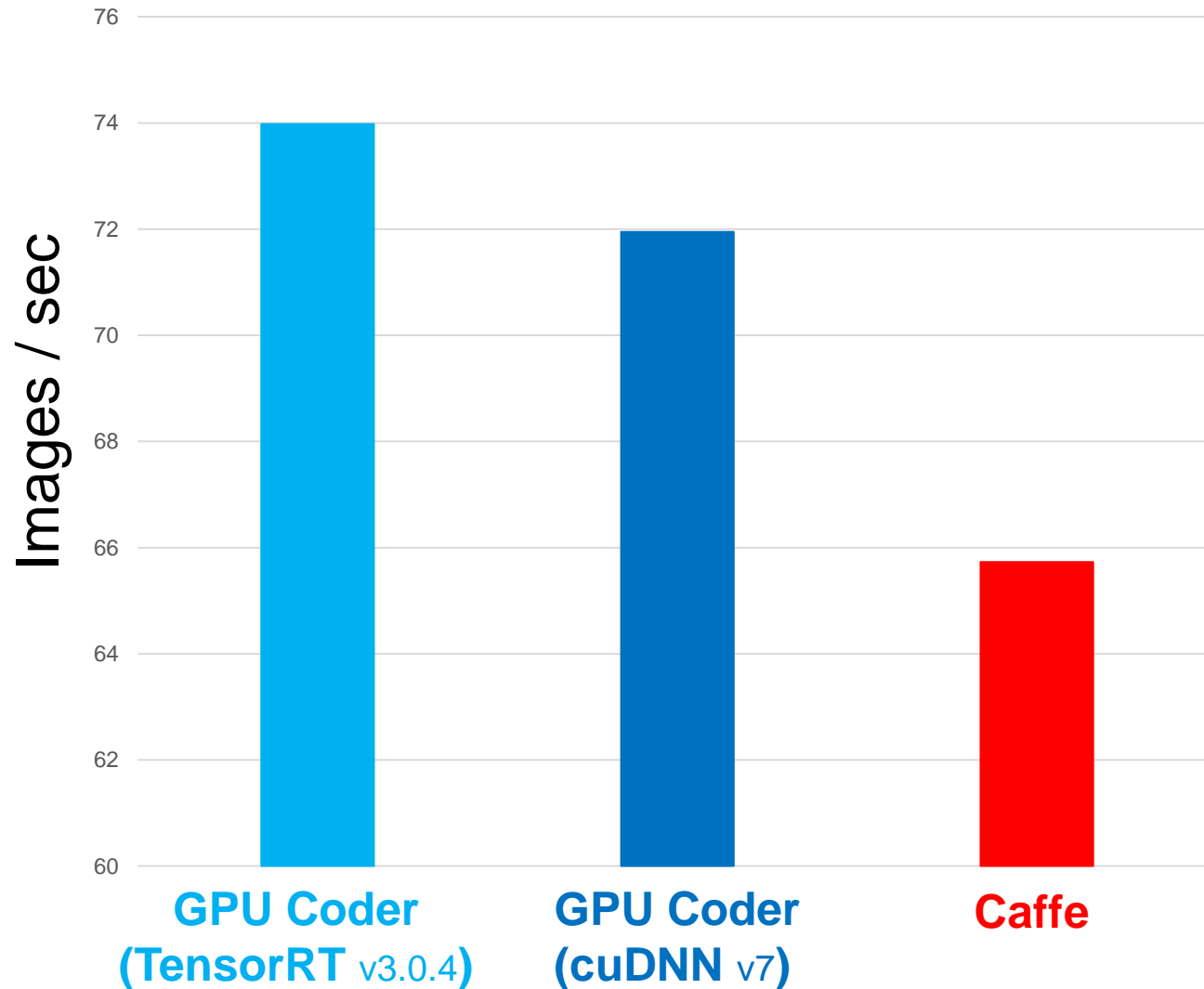
Single Image Prediction (TitanXP GPU, Linux)



TensorFlow
 MXNet
GPU Coder

Single image prediction using Intel® Xeon® CPU - 3.6 GHz, NVIDIA libraries: CUDA8 - cuDNN 7
 TensorFlow 1.6.0, MXNet 1.1.0, MATLAB 18a

Embedded GPU Benchmarking: Jetson TX2



Algorithm Design to Embedded Deployment Workflow

```

% Example MATLAB code snippet
function [y] = myGPUFunction(x)
    % MATLAB algorithm (functional reference)
    % ...
end
    
```

MATLAB algorithm (functional reference)

GPU Coder

Build type

Call CUDA from MATLAB directly

Call CUDA from (C++) hand-coded main()

Call CUDA from (C++) hand-coded main().

.mex

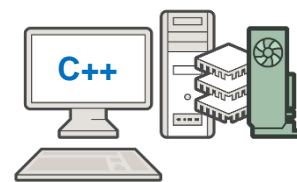
.lib

Cross-compiled .lib

Desktop GPU

Desktop GPU

Embedded GPU



1 Functional test

2 Deployment unit-test

3 Deployment integration-test

4 Real-time test

Demo: Alexnet Deployment with 'mex' Code Generation

The screenshot shows the MATLAB R2017b environment. The current folder is `/tmp/webcamtt`. The file explorer on the left lists the following files:

- test_alexnet_codegen.m
- synsetWords.txt
- peppers_out.png
- peppers.png
- old_workspace.mat
- getAlexnet.m
- cleanup.m
- alexnet_webcam.m
- alexnet_predict.prj
- alexnet_predict.m
- alexnet_live.m

The Command Window on the right displays the following text:

```
fx >>
```

The workspace table is currently empty:

Name	Value

Algorithm Design to Embedded Deployment on Tegra GPU

```

%T_Coder1: Program Reference: T_C1
1. Program Reference: T_C1
2. Program Reference: T_C1
3. Program Reference: T_C1
4. Program Reference: T_C1
5. Program Reference: T_C1
6. Program Reference: T_C1
7. Program Reference: T_C1
8. Program Reference: T_C1
9. Program Reference: T_C1
10. Program Reference: T_C1
11. Program Reference: T_C1
12. Program Reference: T_C1
13. Program Reference: T_C1
14. Program Reference: T_C1
15. Program Reference: T_C1
16. Program Reference: T_C1
17. Program Reference: T_C1
18. Program Reference: T_C1
19. Program Reference: T_C1
20. Program Reference: T_C1
21. Program Reference: T_C1
22. Program Reference: T_C1
23. Program Reference: T_C1
24. Program Reference: T_C1
25. Program Reference: T_C1
26. Program Reference: T_C1
27. Program Reference: T_C1
28. Program Reference: T_C1
29. Program Reference: T_C1
30. Program Reference: T_C1
31. Program Reference: T_C1
32. Program Reference: T_C1
33. Program Reference: T_C1
34. Program Reference: T_C1
35. Program Reference: T_C1
36. Program Reference: T_C1
37. Program Reference: T_C1
38. Program Reference: T_C1
39. Program Reference: T_C1
40. Program Reference: T_C1
41. Program Reference: T_C1
42. Program Reference: T_C1
43. Program Reference: T_C1
44. Program Reference: T_C1
45. Program Reference: T_C1
46. Program Reference: T_C1
47. Program Reference: T_C1
48. Program Reference: T_C1
49. Program Reference: T_C1
50. Program Reference: T_C1
51. Program Reference: T_C1
52. Program Reference: T_C1
53. Program Reference: T_C1
54. Program Reference: T_C1
55. Program Reference: T_C1
56. Program Reference: T_C1
57. Program Reference: T_C1
58. Program Reference: T_C1
59. Program Reference: T_C1
60. Program Reference: T_C1
61. Program Reference: T_C1
62. Program Reference: T_C1
63. Program Reference: T_C1
64. Program Reference: T_C1
65. Program Reference: T_C1
66. Program Reference: T_C1
67. Program Reference: T_C1
68. Program Reference: T_C1
69. Program Reference: T_C1
70. Program Reference: T_C1
71. Program Reference: T_C1
72. Program Reference: T_C1
73. Program Reference: T_C1
74. Program Reference: T_C1
75. Program Reference: T_C1
76. Program Reference: T_C1
77. Program Reference: T_C1
78. Program Reference: T_C1
79. Program Reference: T_C1
80. Program Reference: T_C1
81. Program Reference: T_C1
82. Program Reference: T_C1
83. Program Reference: T_C1
84. Program Reference: T_C1
85. Program Reference: T_C1
86. Program Reference: T_C1
87. Program Reference: T_C1
88. Program Reference: T_C1
89. Program Reference: T_C1
90. Program Reference: T_C1
91. Program Reference: T_C1
92. Program Reference: T_C1
93. Program Reference: T_C1
94. Program Reference: T_C1
95. Program Reference: T_C1
96. Program Reference: T_C1
97. Program Reference: T_C1
98. Program Reference: T_C1
99. Program Reference: T_C1
100. Program Reference: T_C1
    
```

MATLAB algorithm
(functional reference)

GPU Coder

Build type

Call CUDA
from MATLAB
directly

.mex

Call CUDA from
(C++) hand-
coded main()

.lib

Call CUDA from (C++)
hand-coded main().
Cross-compiled on host
with Linaro toolchain

Cross-compiled
.lib

Tesla
GPU

Tesla
GPU

Tegra GPU

1 Functional test

2 Deployment
unit-test

3 Deployment
integration-test

4 Real-time test

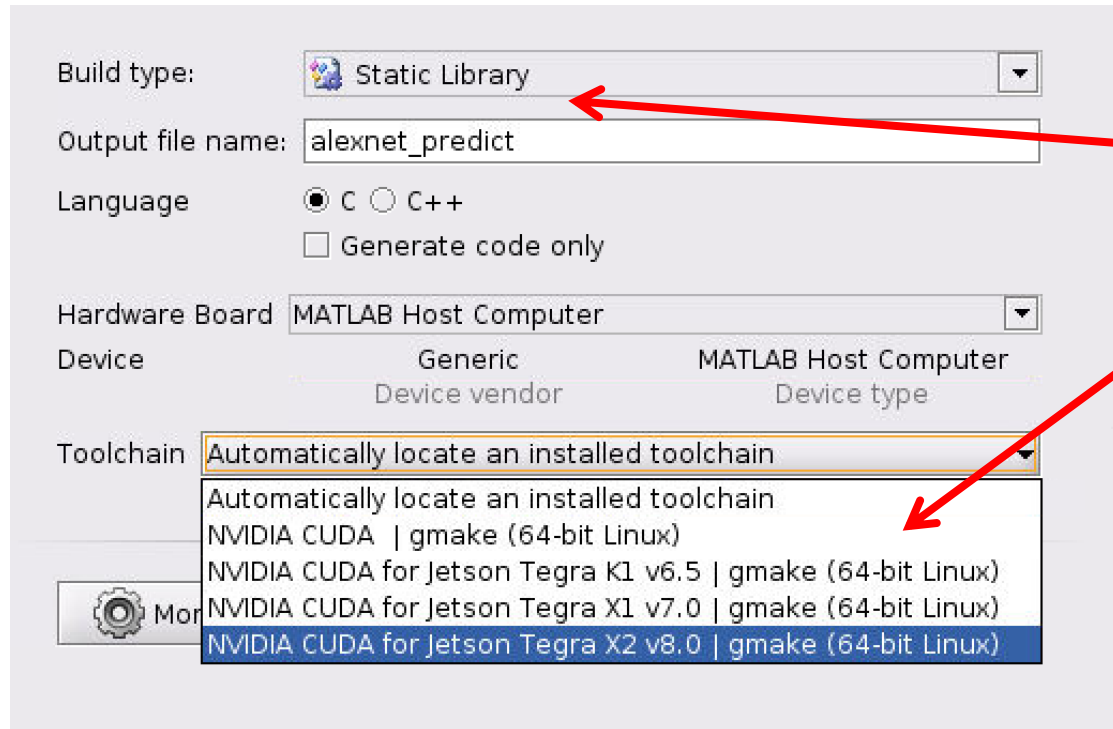
(Test in MATLAB on host)
MATLAB EXPO 2018

(Test generated code in
MATLAB on host + GPU)

(Test generated code within
C/C++ app on host + GPU)

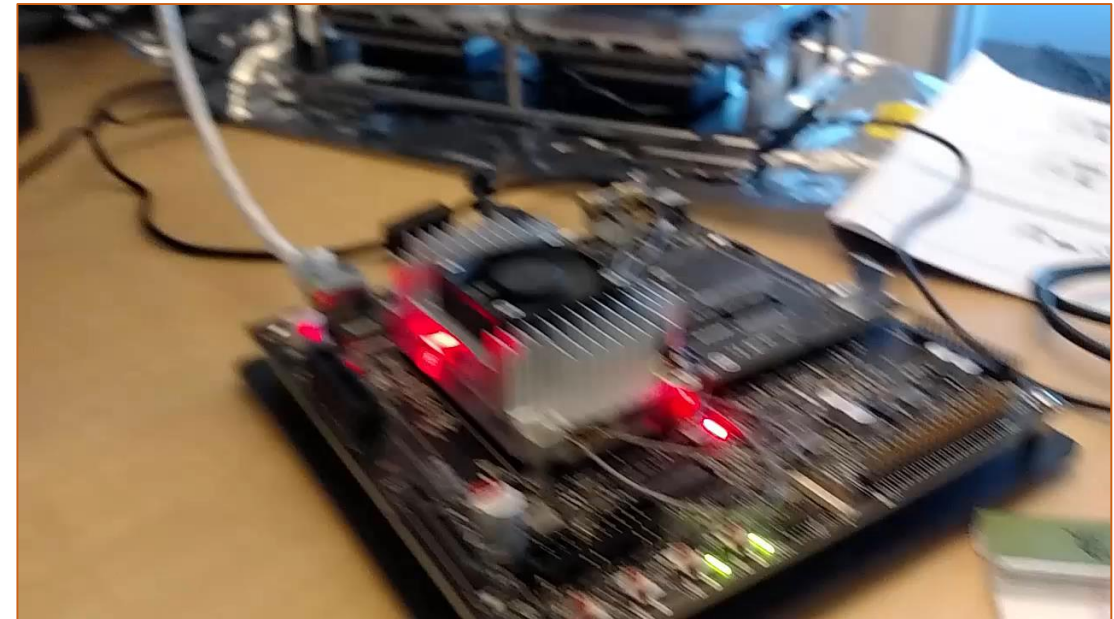
(Test generated code within
C/C++ app on Tegra target)

Alexnet Deployment to Tegra: Cross-Compiled with 'lib'

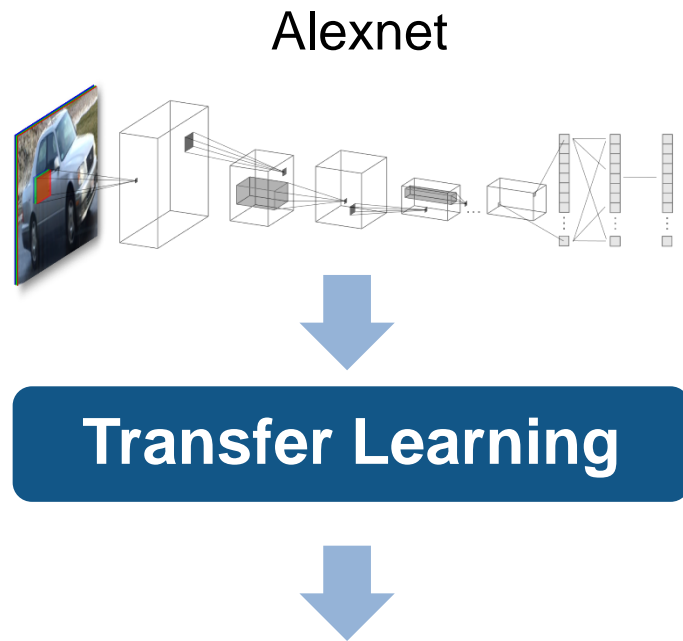


Two small changes

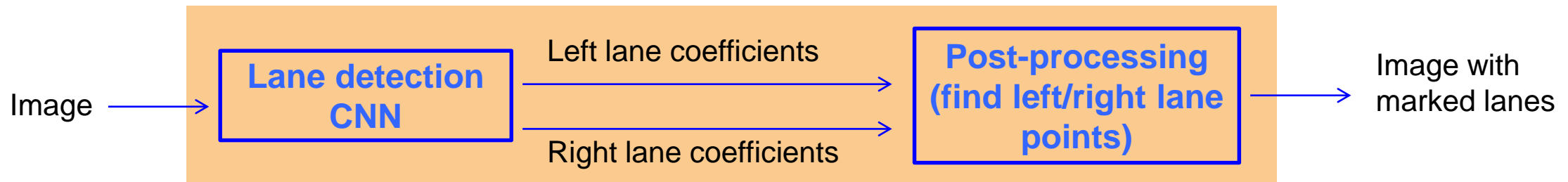
1. Change build-type to 'lib'
2. Select cross-compile toolchain



End-to-End Application: Lane Detection



Output of CNN is lane parabola coefficients according to: $y = ax^2 + bx + c$



GPU coder generates code for whole application

Deep learning workflow in MATLAB

