



모델 기반 설계를 활용한 인공지능 시스템 개발

YouTube Live Webinar

2021-10-27

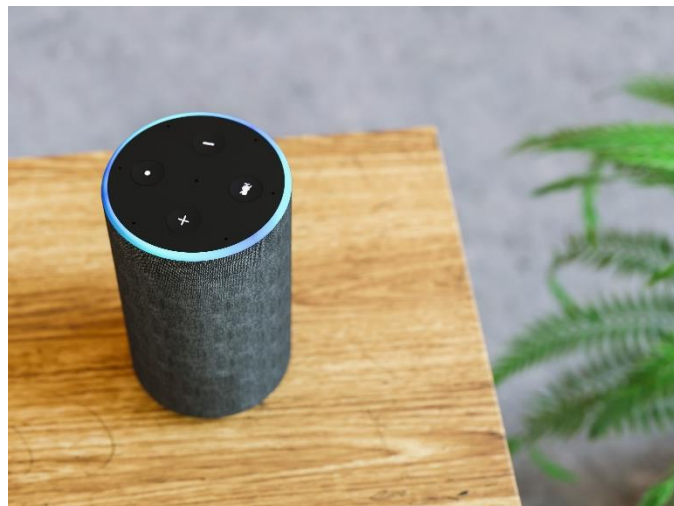
송완빈 과장

Application Engineer @ MathWorks

wsong@mathworks.com



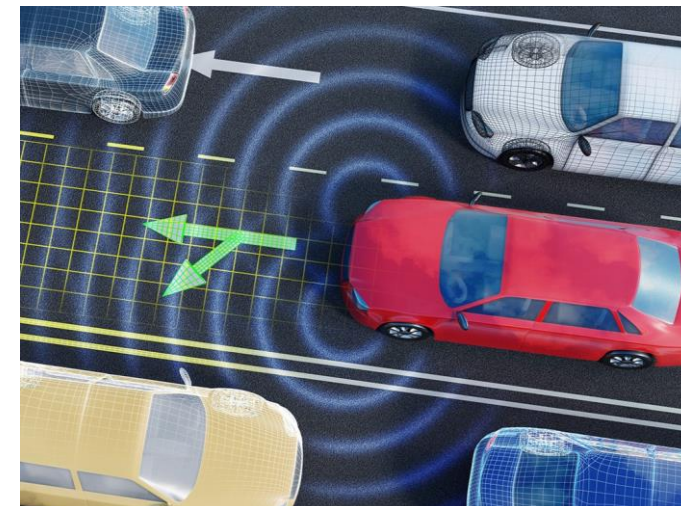
Deep learning is part of our everyday lives



Speech Recognition



Face Detection



Automated Driving

Deep Learning in MATLAB used in Industry

BMW



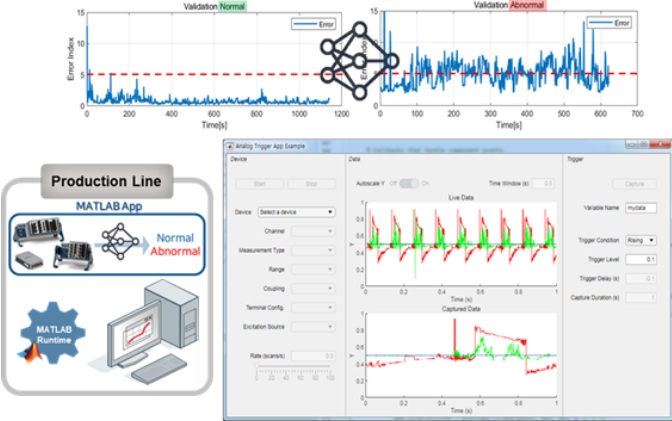
Oversteering Detection

Atlas Copco



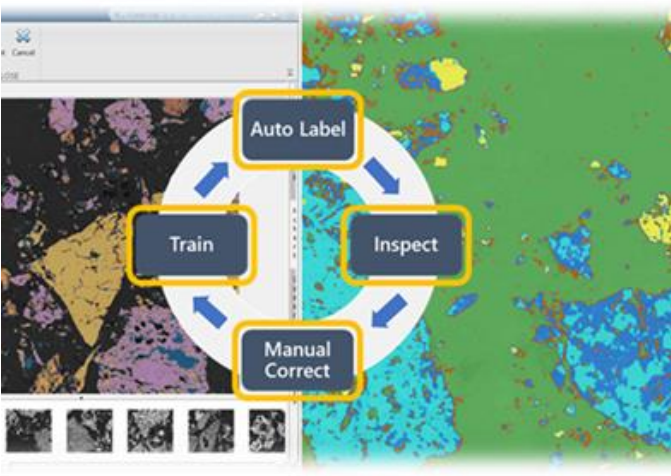
Digital Twins of Compressors

LG Energy Solution



Condition monitoring system

Hyundai Steel



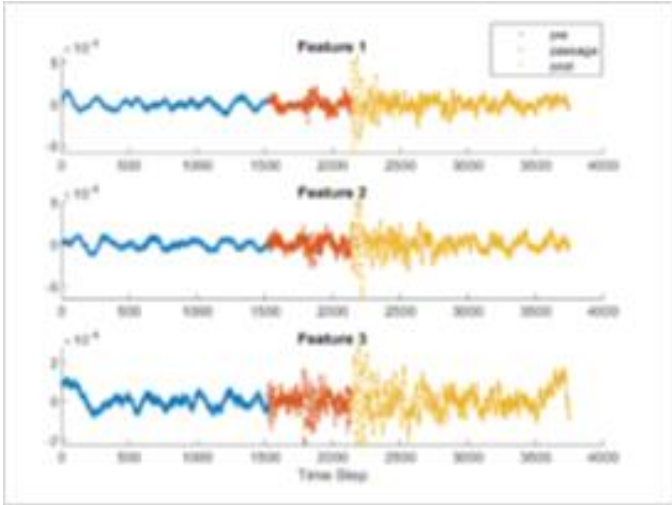
Steel Image Analysis

AirBus



Defect Detection

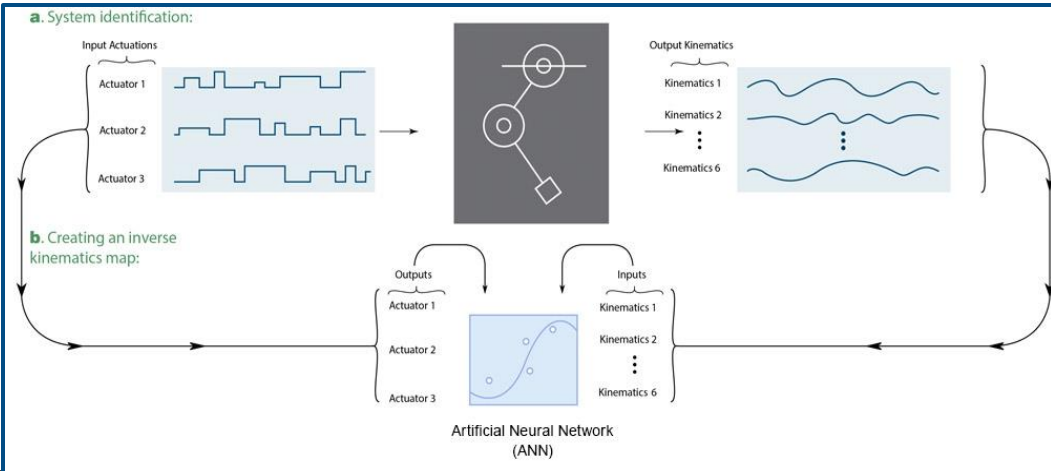
Shell



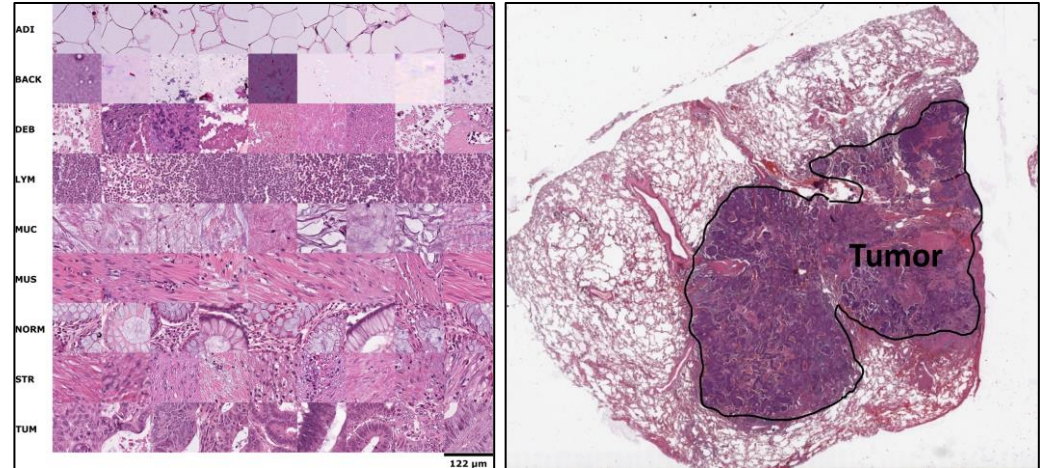
Seismic Event Detection

Deep Learning and AI in Research

University of Southern California



Reinforcement Learning for Robotic Arm



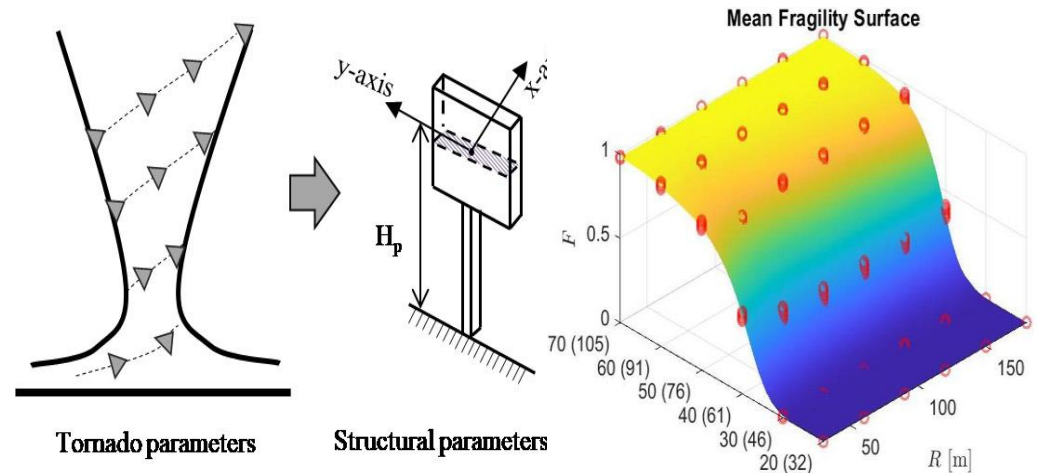
Deep Learning for Tumor Detection

DKFZ
Heidelberg

University of Twente



Augmented Reality of blood flow




Neural Networks simulate tornadic wind load


Northeastern
University

AI-driven system design

Data Preparation

 Data cleansing and preparation

 Human insight

 Simulation-generated data

AI Modeling

 Model design and tuning


 Hardware accelerated training

 Interoperability

Simulation & Test

 Integration with complex systems

 System simulation

 System verification and validation

Deployment

 Embedded devices

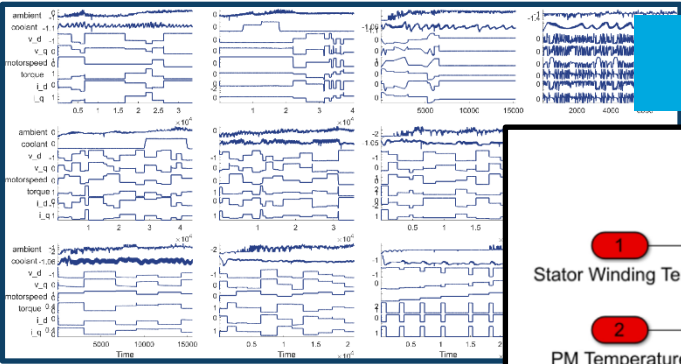
 Enterprise systems

 Edge, cloud, desktop

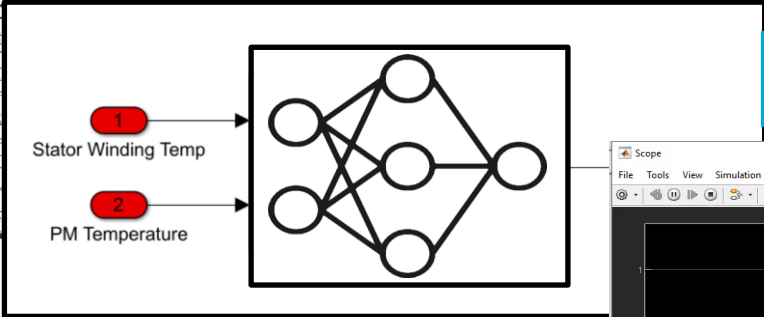
Deep learning for signals

Deep learning for motor temperature estimation

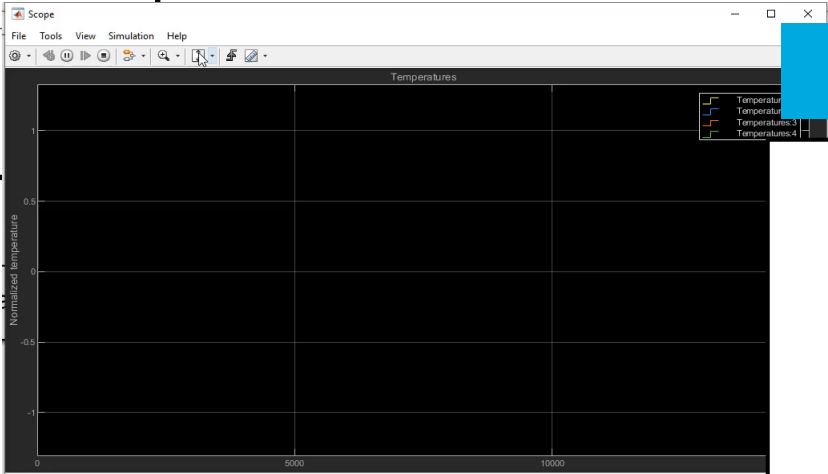
Data Preparation



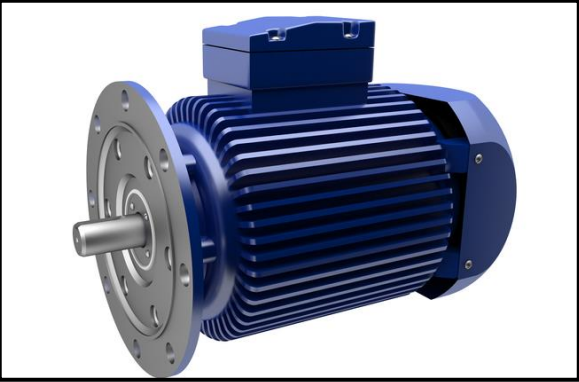
AI Modeling



Simulation & Test





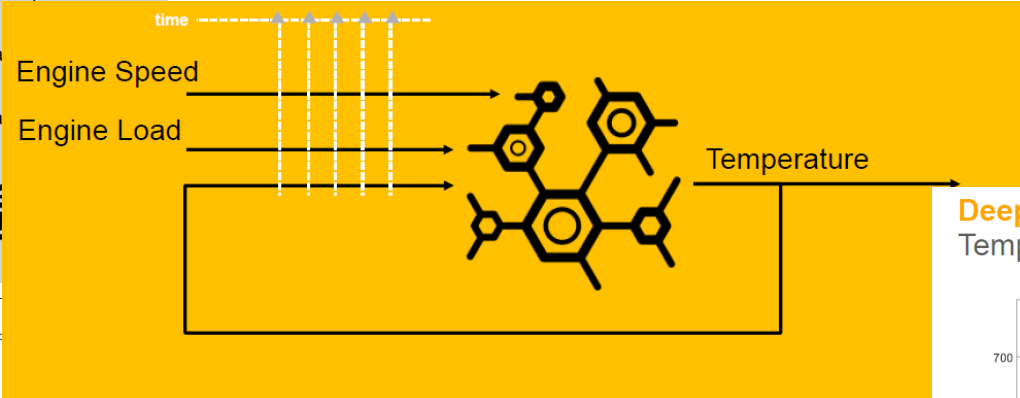
Deployment



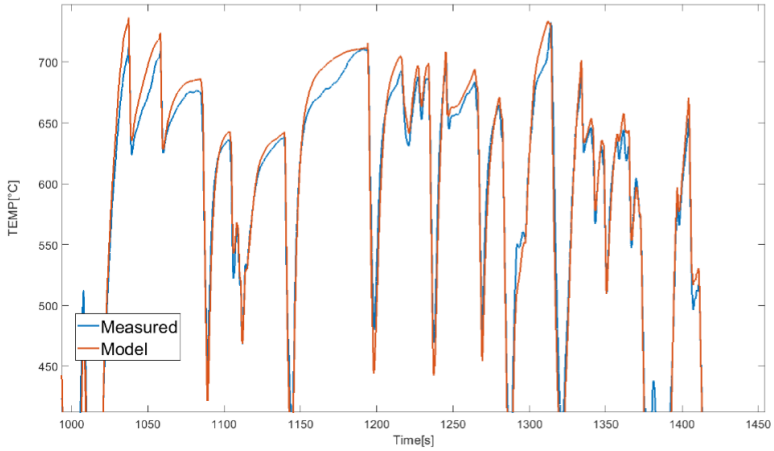
Continental uses data-driven engine temperature models for ECU development

Classical ECU Functions Advantages/Disadvantages

Advantages	Disadvantage
<p>Physically motivated</p> <p>High understanding of whats going on (intermediate signals have typically physical units)</p> <p>Enabling "transfer learning" for single HW change</p> 	<ul style="list-style-type: none"> › Require development (modelling + coding) › Require methodology development for calibration = training › Require tooling for the tra (backpropagation) › Require very special mea engine test bench 



Deep Dynamical Systems
Temperature example 40min of driving (validation)



[Link to presentation](#)

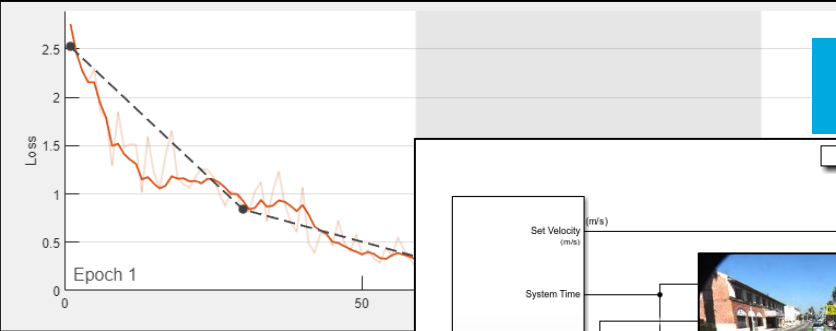
Deep learning for images and video

Deep learning for vehicle detection

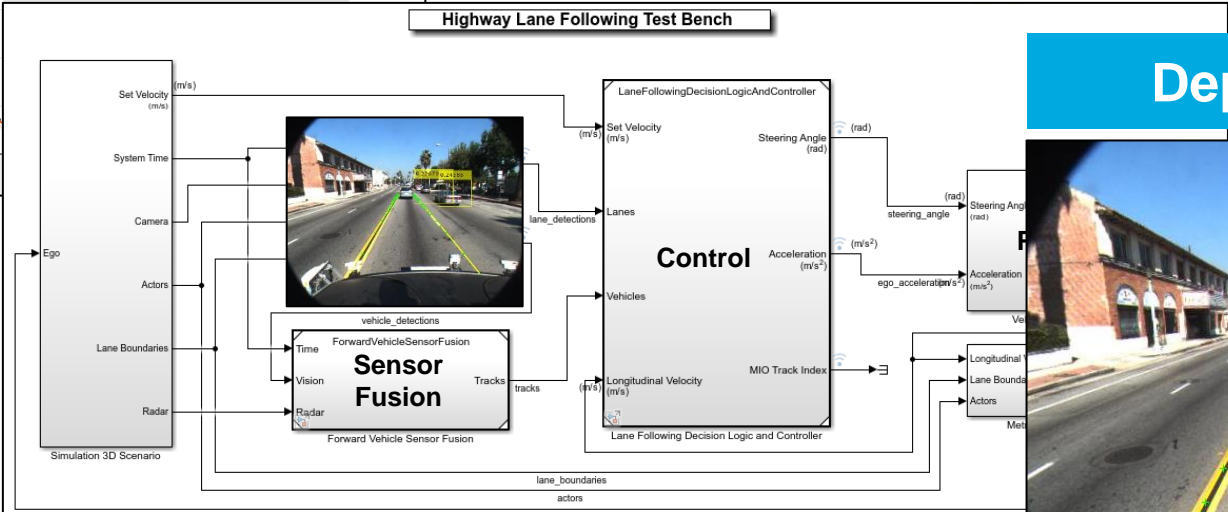
Data Preparation



AI Modeling



Simulation & Test

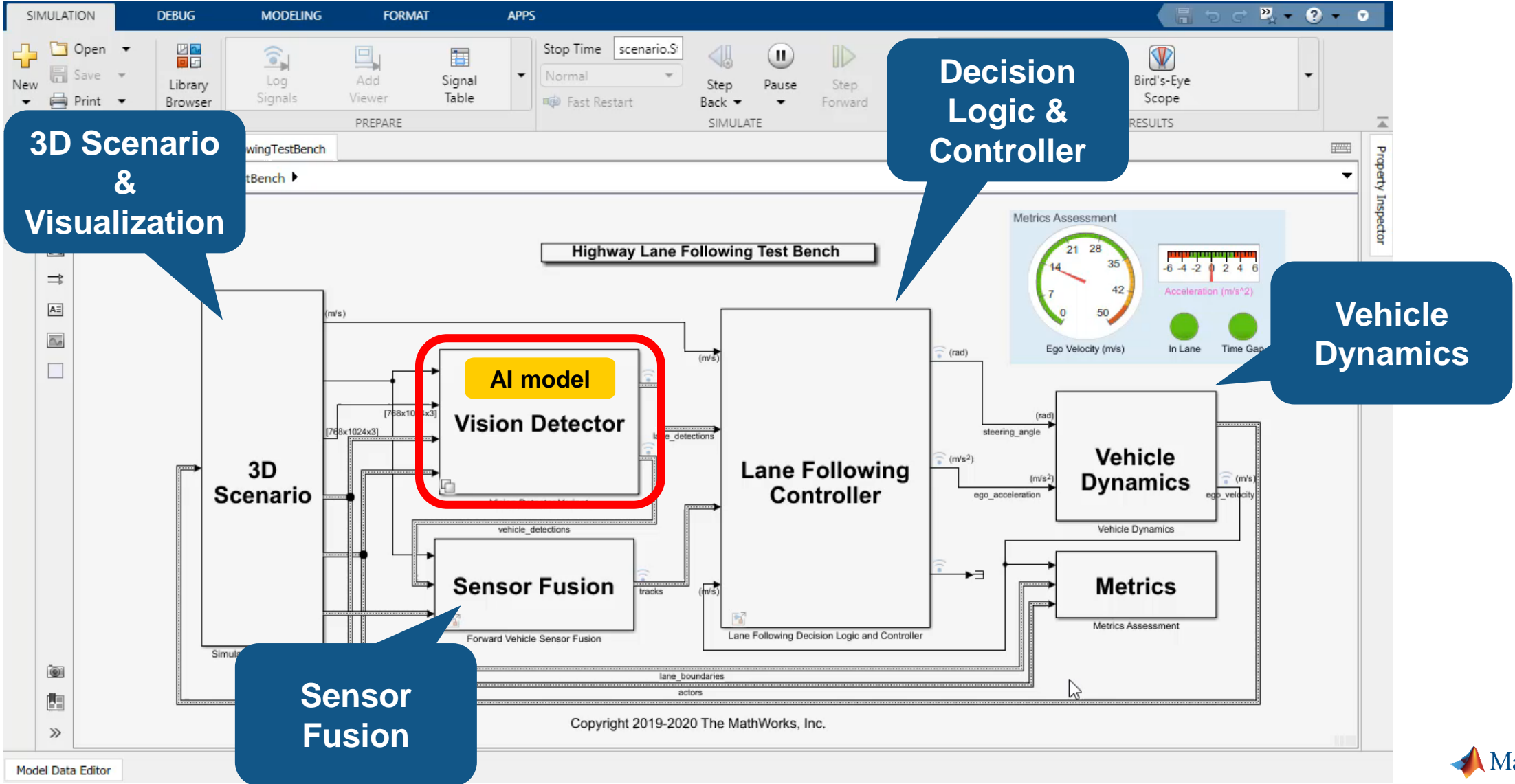


Deployment

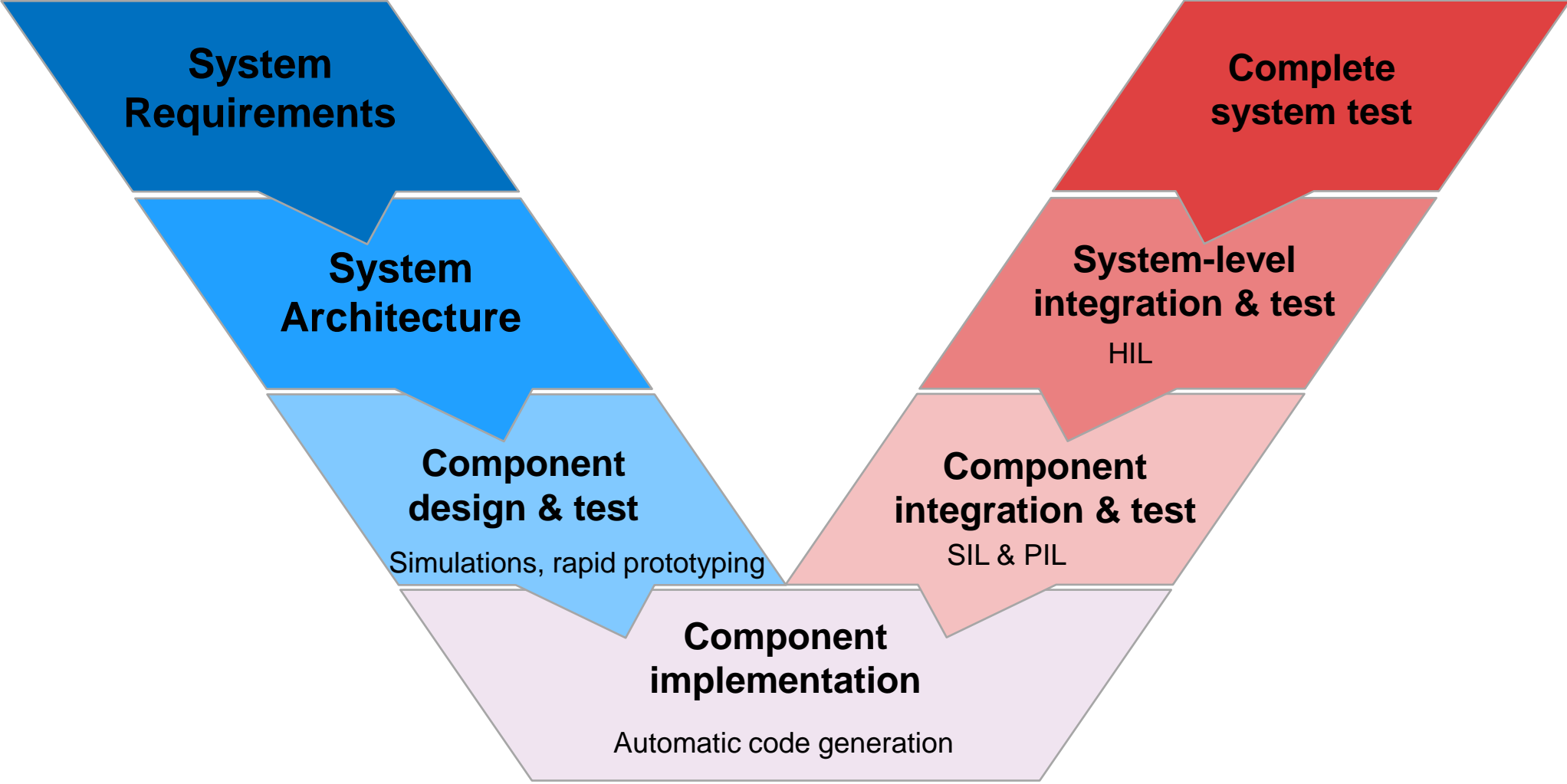


Deep learning is often part of a larger system

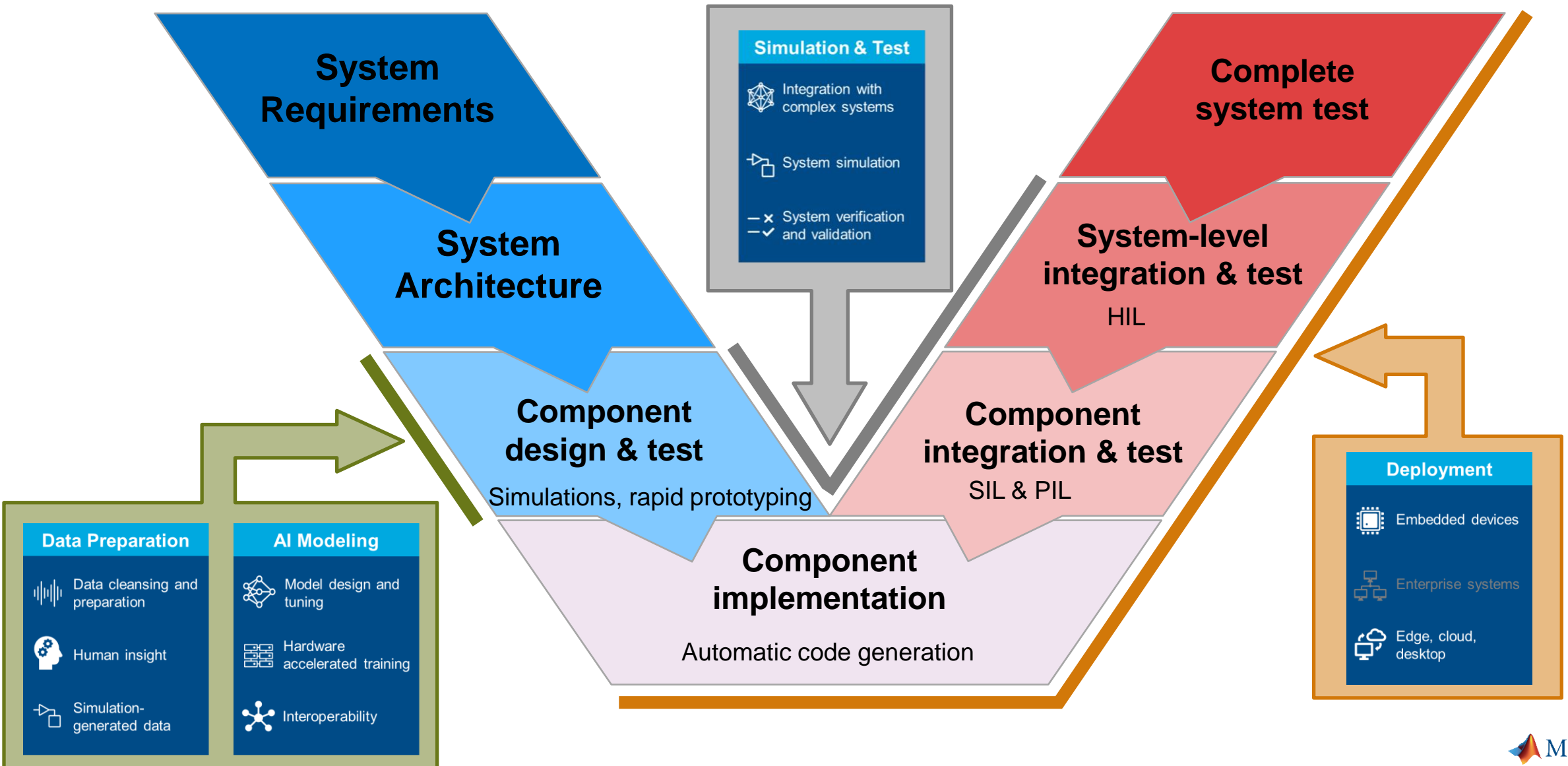
Simulate and test all your components together in Simulink



Development workflow with Model-Based Design



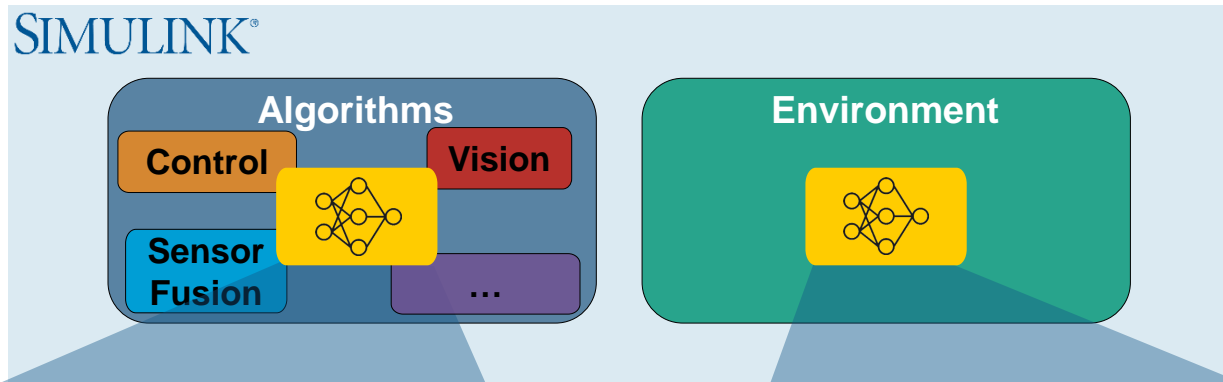
AI system development with Model-Based Design



What can you do with AI?

- **Is the model slow?** Not possible to get real time for HIL?
 - Speed up high fidelity models by creating AI based surrogate models
- **Is the model inaccurate** for certain outputs compared to lab data?
 - Increase accuracy of complex processes, like emission calculations, by using data driven models trained on measured lab data
- **Is there a sensor value you would like to have** but that you can only afford to get in labs?
 - Create virtual sensors using AI
- Many more...

Integrating deep learning models into Simulink



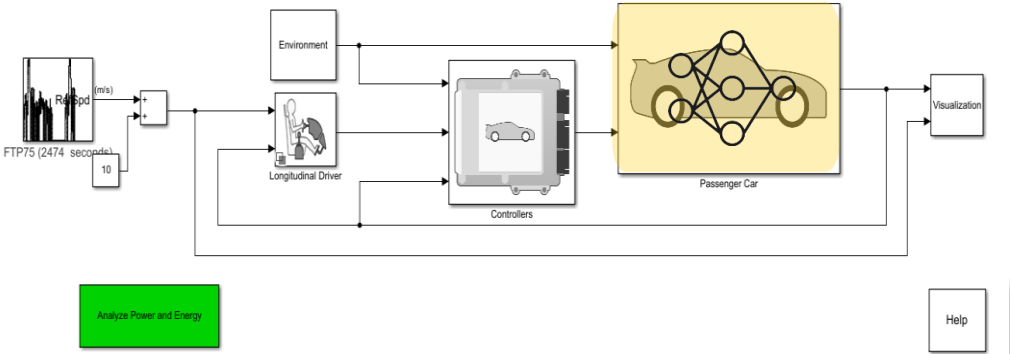
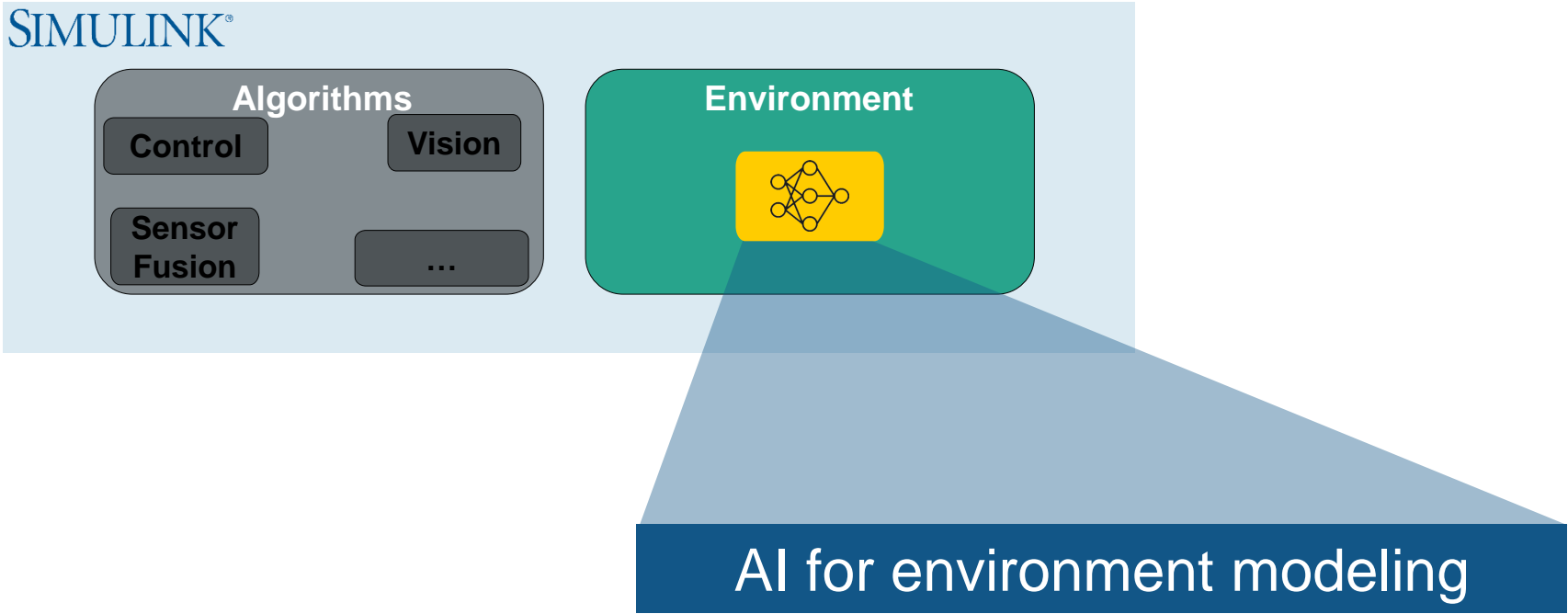
AI for algorithm development

- Simulate for system-level testing
- Verify system requirements
- Deploy to CPU/GPU/ECU

AI for environment modeling

- Speed up high-fidelity Simulink model
- Reduce Simulink model complexity
- Enable HIL tests if first-principles model cannot be obtained

Deep learning for environment modeling



Deep Learning for engine surrogate model

Physics-based vs data-driven modeling

First principles models and data-driven models can co-exist

First-principles models

Physics, math, domain knowledge

Data-driven models

Statistics, optimization, AI

White-box

Gray-box

Black-box

Advantages

- May capture (global) parameterizable behaviors with low/high fidelity
- Have clear (explainable) physical meaning
- Do not require data engineering

Challenges

- Can be challenging/impossible to derive
- Require significant time for derivation
- Require expertise in the respective domain

Advantages

- May succeed when first-principles models are unavailable or challenging/impossible to find
- May reduce complexity, simulate faster
- Can leverage existing, measured data
- Do not require domain knowledge

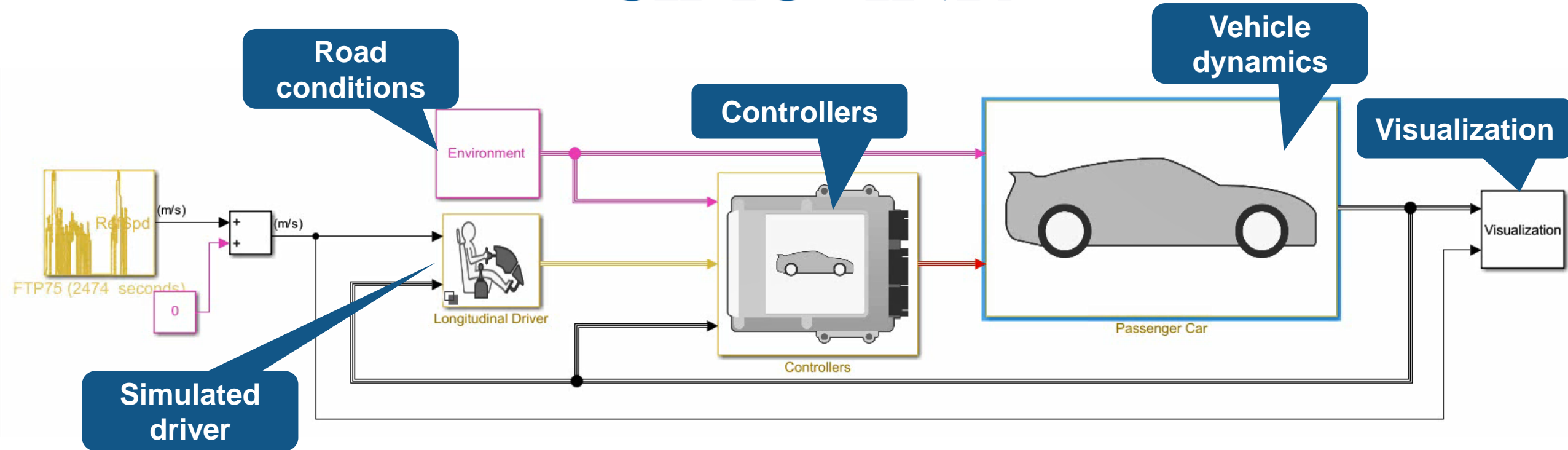
Challenges

- Require a lot of data
- Are often not
 - interpretable, explainable
 - easily parameterizable in a physically meaningful way
- Cannot extrapolate well beyond training data

Example overview

Replacing a first-principles engine model with an AI-based surrogate

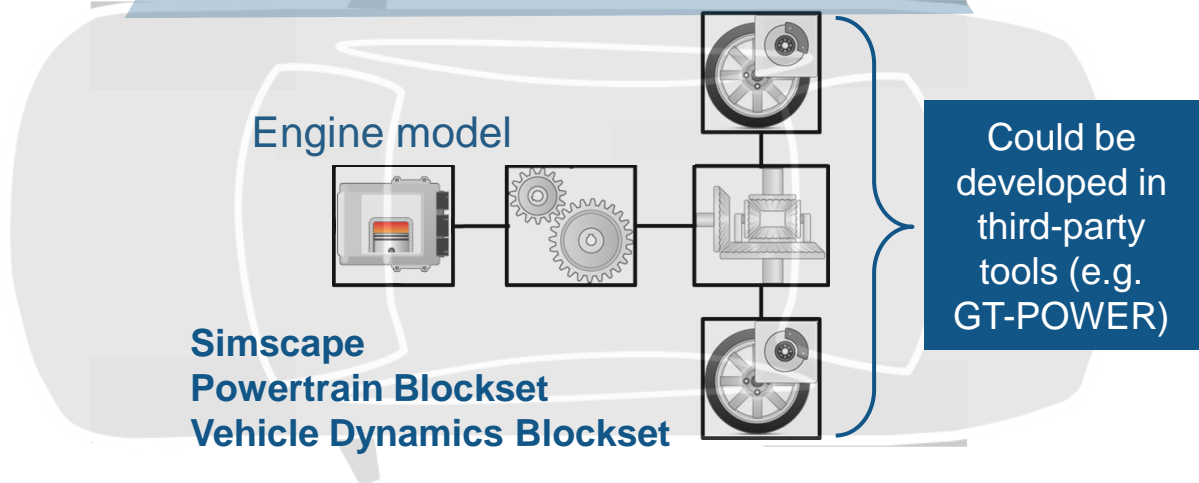
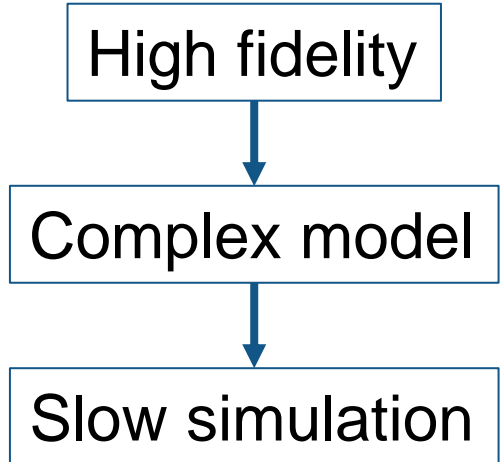
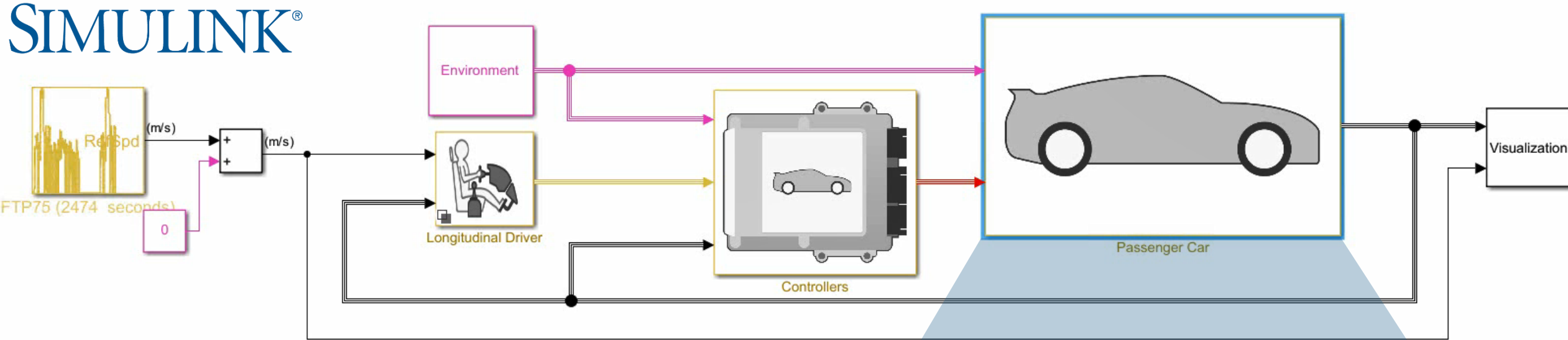
SIMULINK®



Closed-loop control of vehicle speed

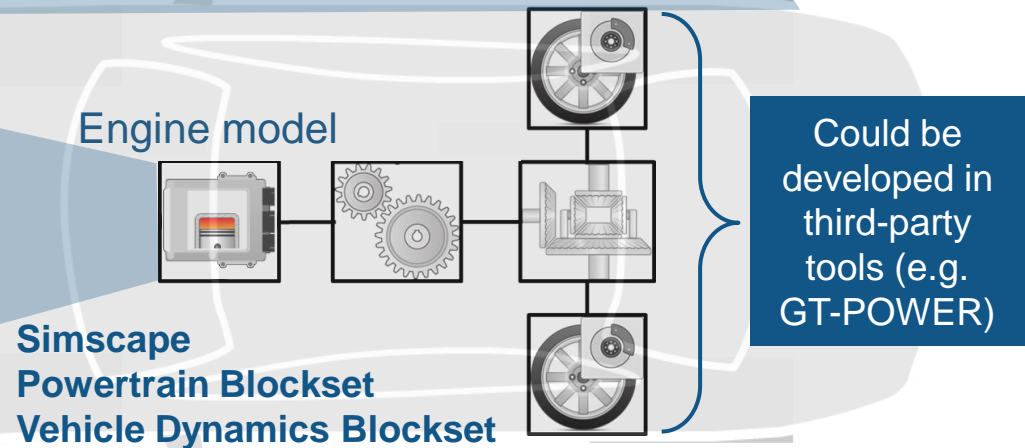
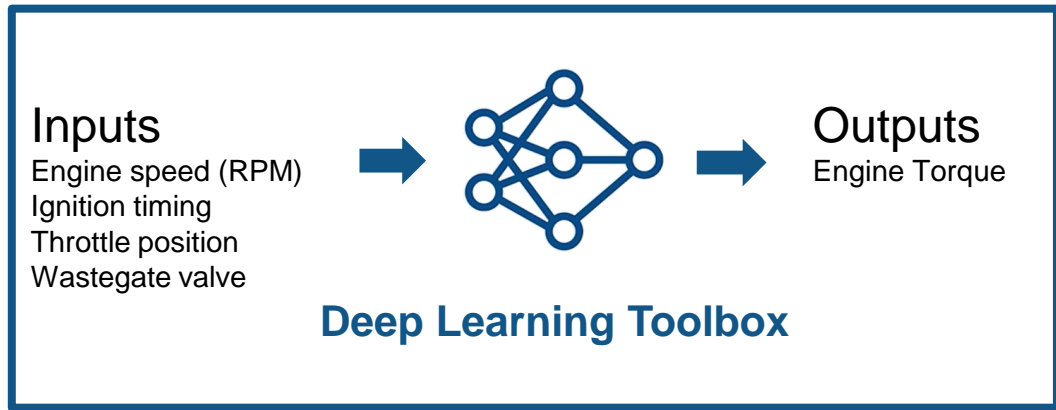
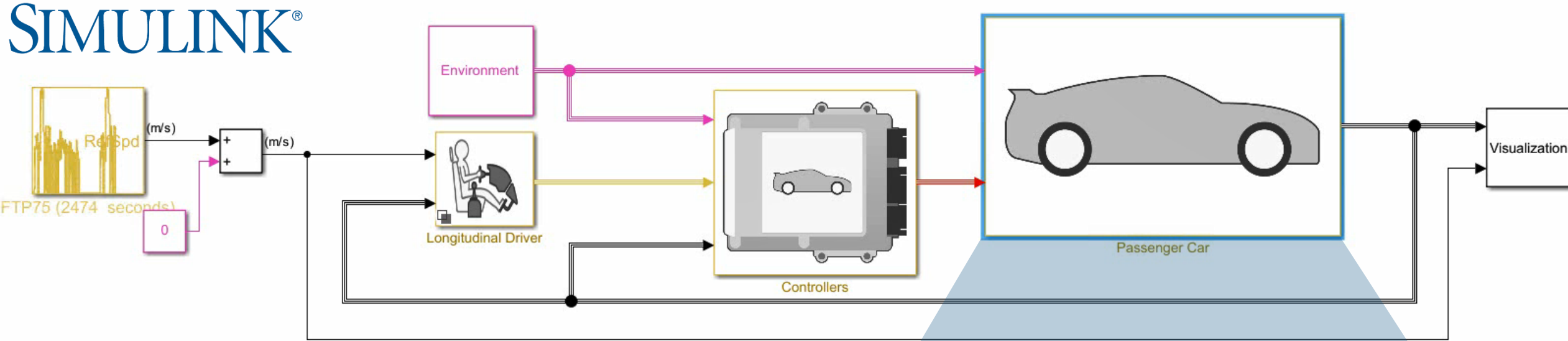
Example overview

Replacing a first-principles engine model with an AI-based surrogate



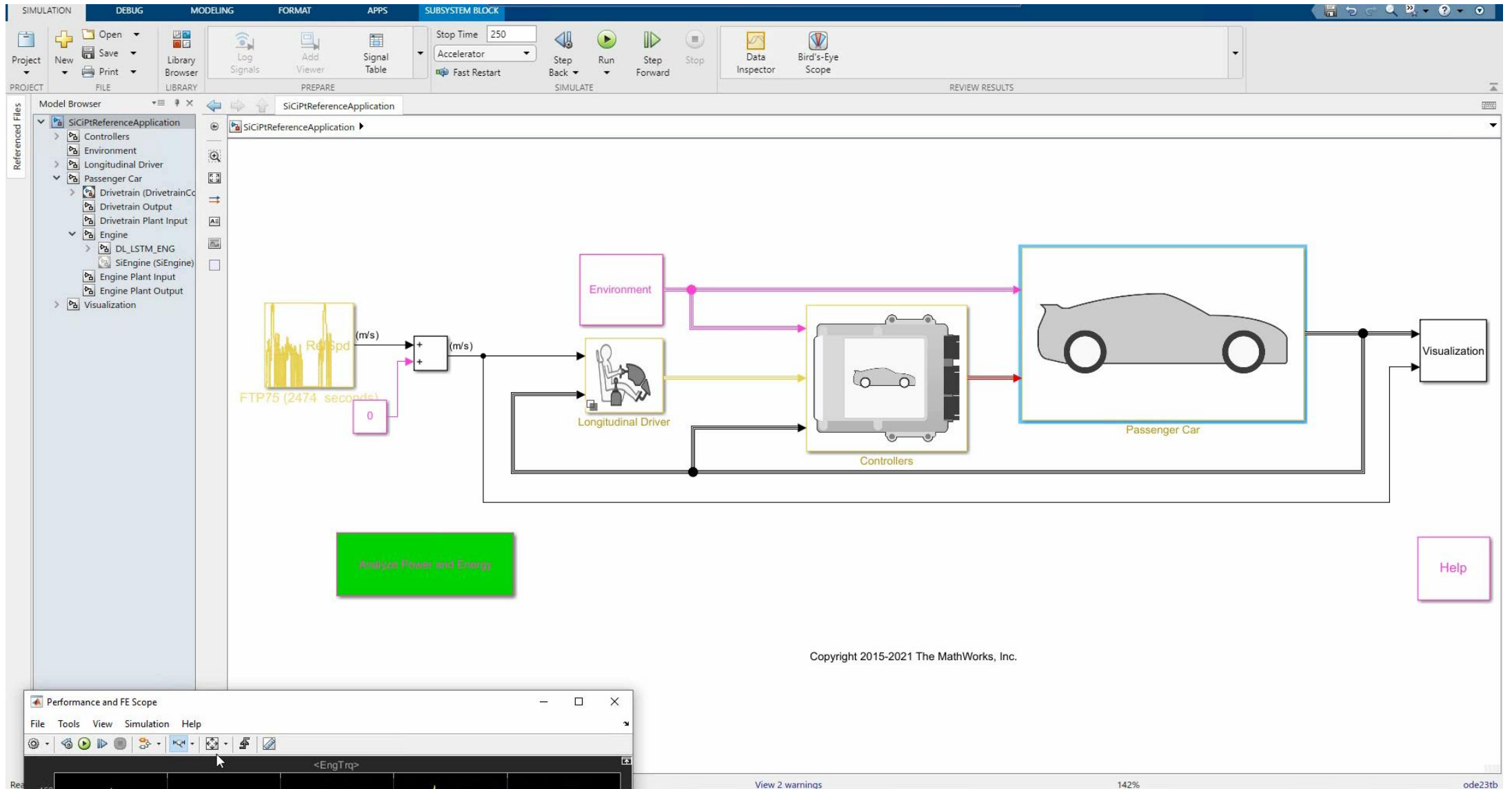
Example overview

Replacing a first-principles engine model with an AI-based surrogate




Example overview

System-level simulation




AI-driven system design

Data Preparation

 Data cleansing and preparation

 Human insight

 Simulation-generated data

AI Modeling

 Model design and tuning


 Hardware accelerated training

 Interoperability

Simulation & Test

 Integration with complex systems

 System simulation

 System verification and validation

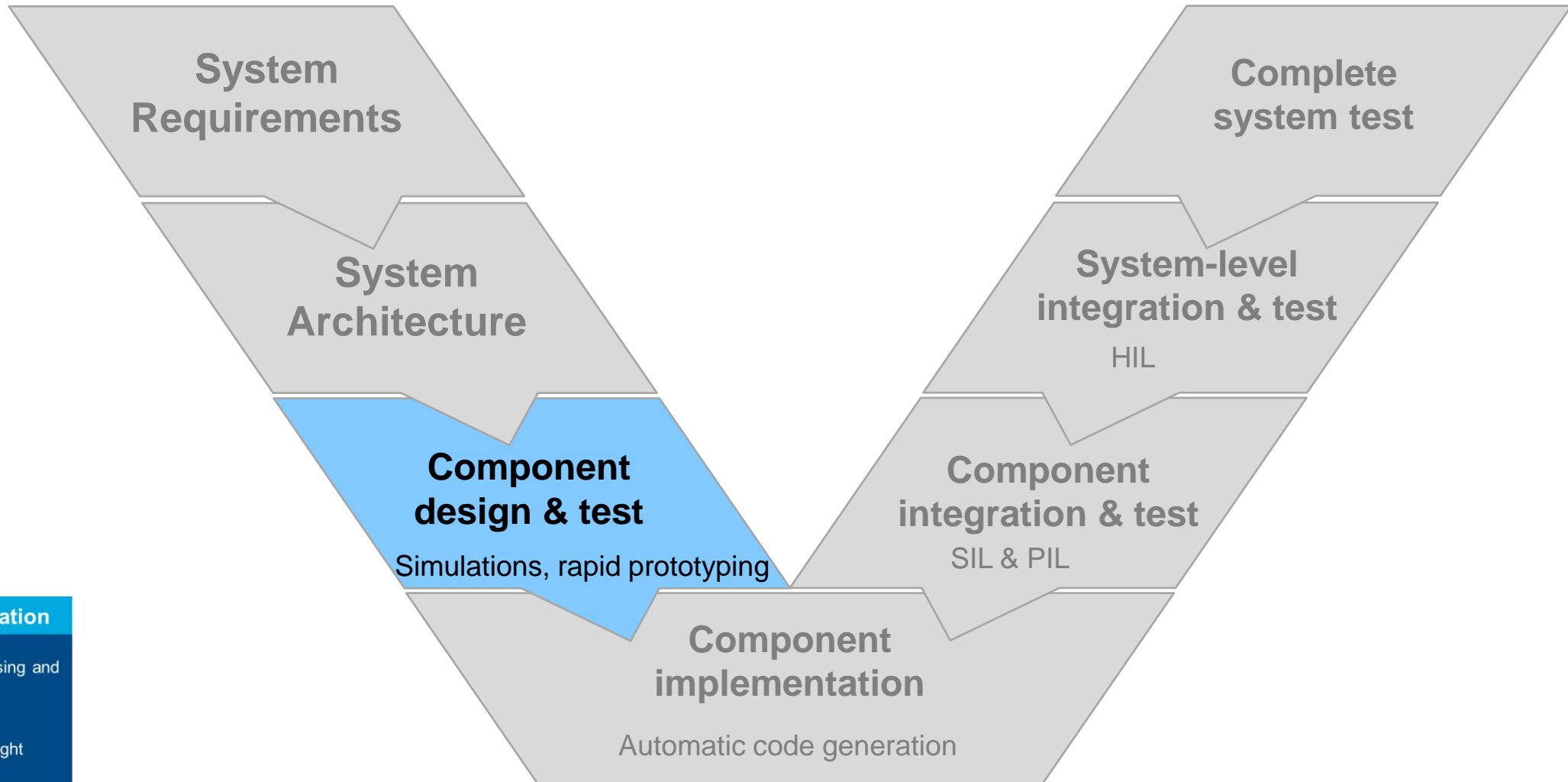
Deployment

 Embedded devices

 Enterprise systems

 Edge, cloud, desktop

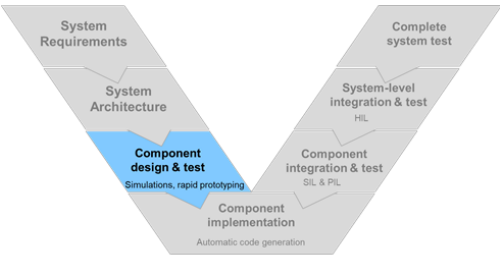
AI system development with Model-Based Design



Data Preparation

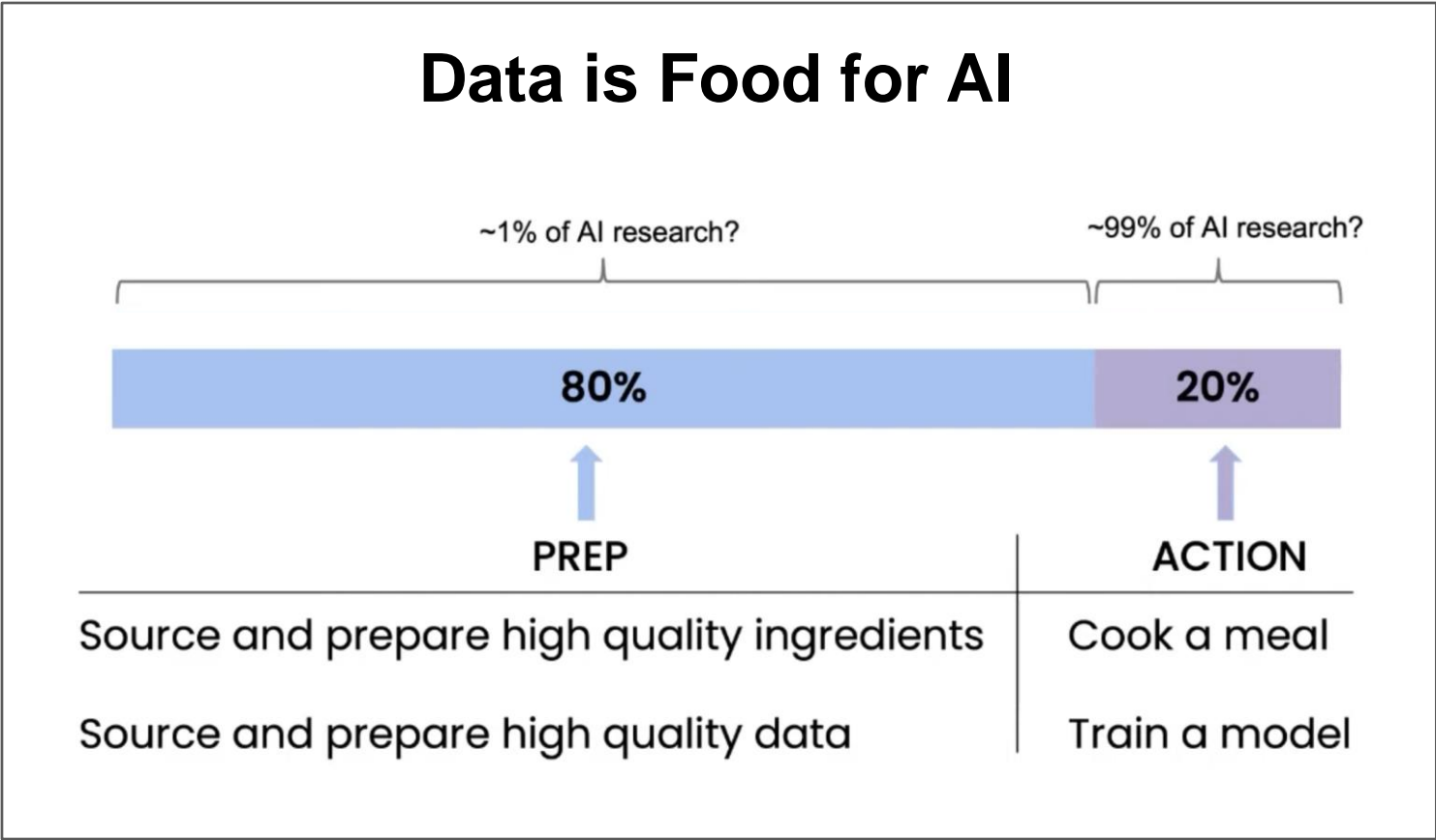
-  Data cleansing and preparation
-  Human insight
-  Simulation-generated data

Data preparation represents most of your AI effort



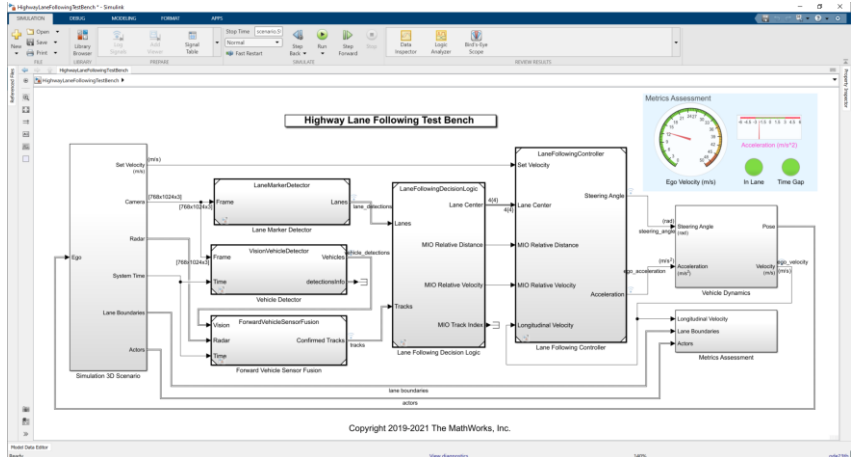
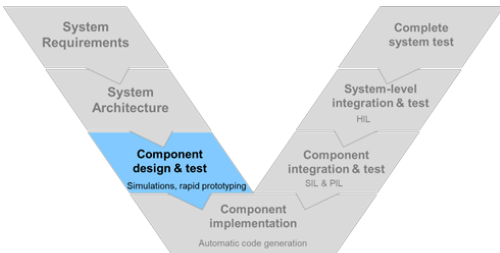
Data Preparation

- Data cleansing and preparation
- Human insight
- Simulation-generated data



Source: Andrew NG slide from MLOps 2021

Generate synthetic data for training



Simulink/Simscape



GANs

Data Preparation

- Data cleansing and preparation
- Human insight
- Simulation-generated data

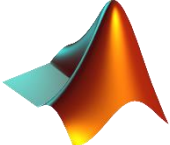


Unreal Engine®

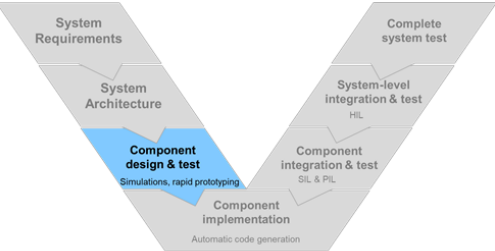
The screenshot shows the 'Wireless Waveform Generator - Spectrum Analyzer' interface. It includes a 'Time Scope' plot showing a complex waveform over time, a 'Spectrum Analyzer' plot showing frequency components, and a 'Constellation Diagram' plot showing signal points in the complex plane. The interface also has various configuration options for modulation, filtering, and impairments.

Wireless Waveform Generator MathWorks

Data preparation demo



Open Part 1



Data Preparation

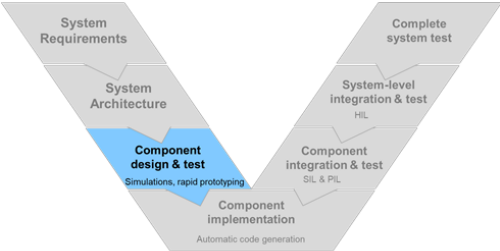
- Data cleansing and preparation
- Human insight
- Simulation-generated data

```

Load data
load LogData20210204_doe512.mat
data = logout;
% extract data which will be used
EngSpd = data{7}.Values.Data';
SpkAdv = data{11}.Values.Data';
ThrPosPct = data{14}.Values.Data';
WgAreaPct = data{15}.Values.Data';
EngTrq_fil = data{4}.Values.Data';
% Catenate 4 feature vectors
X_transient = [ThrPosPct; ... Throttle Position
              WgAreaPct; ... Wastegate valve
              EngSpd; ... Engine speed
              SpkAdv]; % Spark timing
Y_transient = EngTrq_fil; % Engine Torque

Create data index
prepare features and responses
tdatalength = 10000; % The last index of the training data, Data from index 1 to 10000 are used
testdataidx = 100001; % First index of data for testing
% Use indexes to separate training and evaluation data from the overall data.
    
```

Data preparation demo



Data Preparation

Data cleansing and preparation

Human insight

Simulation-generated data

The screenshot shows the MATLAB Live Editor interface. The file explorer on the left lists files: Demo_LSTM_for_EngineTr..., EN_Demo_LSTM_for_Engi..., LogData20210204_doe51..., net.mat, and ReadMe.txt. The code editor displays the following code:

```

load LogData20210204_doe512.mat
data = logargout;
% extract data which will be used
EngSpd = data{7}.Values.Data';
SpkAdv = data{11}.Values.Data';
ThrPosPct = data{14}.Values.Data';
WgAreaPct = data{15}.Values.Data';
EngTrq_fil = data{4}.Values.Data';
% Catenate 4 feature vectors
X_transient = [ThrPosPct; ... Throttle Position
               WgAreaPct; ... Wastegate valve
               EngSpd; ... Engine speed
               SpkAdv]; % Spark timing
Y_transient = EngTrq_fil; % Engine Torque
    
```

Annotations on the screenshot include:

- A green box around the `load` command with the text "Load data".
- A red box around the `X_transient` and `Y_transient` assignments with the text "prepare features and responses".
- A red box around the `tdatalength` and `testdataidx` assignments with the text "Create data index".
- A large green text overlay on the right says "Load data from a '.mat' file".
- A large red text overlay at the bottom right says "prepare features and responses".


The Command Window at the bottom shows the following output:

```


WgAreaPct    1x118962 do...
X_transient  4x118962 do..
Y_transient  1x118962 do..
Ymu         94.1296
Ysig        25.4823
    
```

AI-driven system design

Data Preparation

 Data cleansing and preparation

 Human insight

 Simulation-generated data

AI Modeling

 Model design and tuning

 Hardware accelerated training

 Interoperability

Simulation & Test

 Integration with complex systems

 System simulation

 System verification and validation

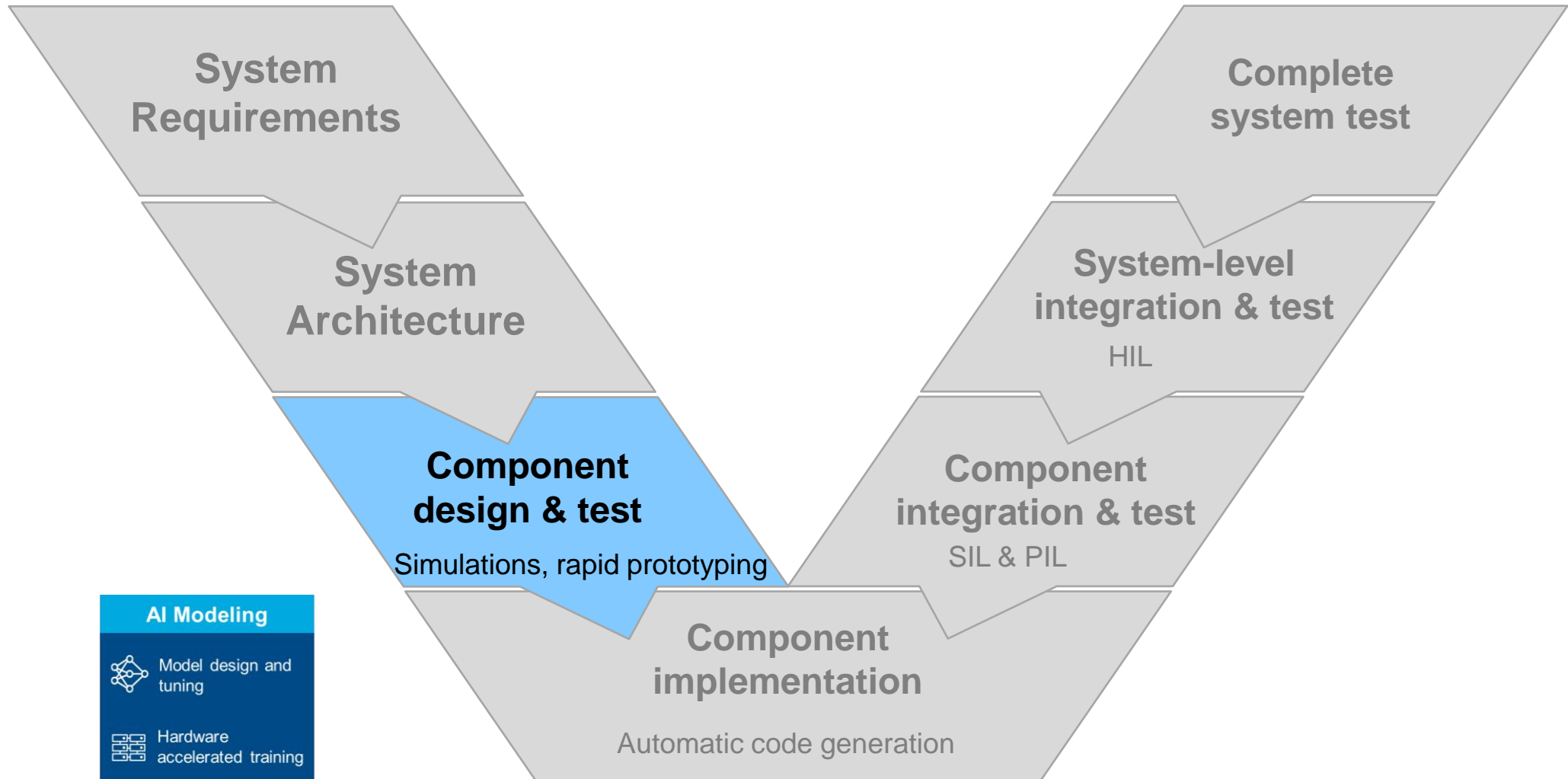
Deployment

 Embedded devices




 Enterprise systems

 Edge, cloud, desktop

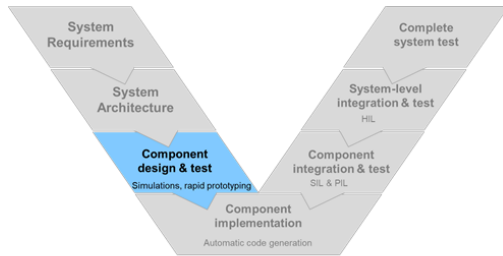
AI system development with Model-Based Design



AI Modeling

-  Model design and tuning
-  Hardware accelerated training
-  Interoperability

Start with a complete set of algorithms and pre-built models



AI Modeling



Model design and tuning



Hardware accelerated training



Interoperability

Algorithms

Machine learning

Trees, Naïve Bayes, SVM...

Deep learning

CNNs, GANs, LSTM, MIMO...

Reinforcement learning

DQN, A2C, DDPG...

Regression

Linear, nonlinear, trees...

Unsupervised learning

K-means, PCA, GMM...

Predictive maintenance

RUL models, condition indicators...

Bayesian optimization

Pre-built models

Image classification models

AlexNet, GoogLeNet, VGG, SqueezeNet, ShuffleNet, ResNet, DenseNet, Inception...

Reference examples

Object detection

Vehicles, pedestrians, faces...

Semantic segmentation

Roadway detection, land cover classification, tumor detection...

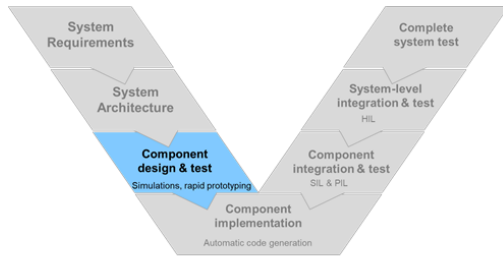
Signal and speech processing

Denoising, music genre recognition, keyword spotting, radar waveform classification...

...and more...

AI modeling demo: Creating and training the LSTM model

Use LSTM networks to capture time dependencies in time-series data

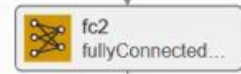
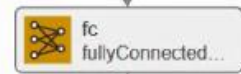
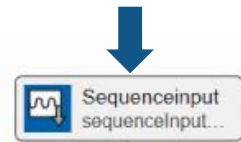


AI Modeling

- Model design and tuning
- Hardware accelerated training
- Interoperability

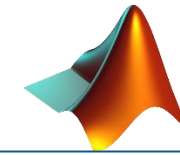
Inputs

Engine speed (RPM)
Ignition timing
Throttle position
Wastegate valve

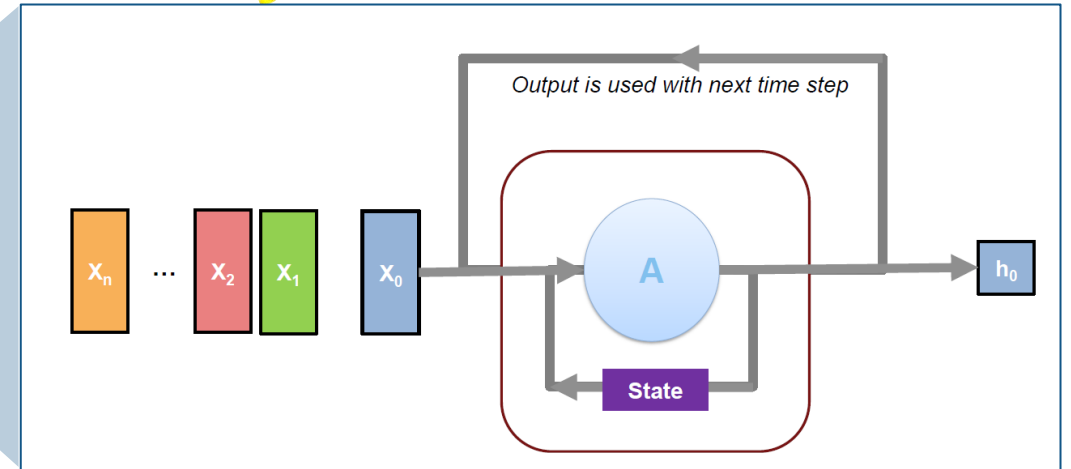


Outputs

Engine Torque



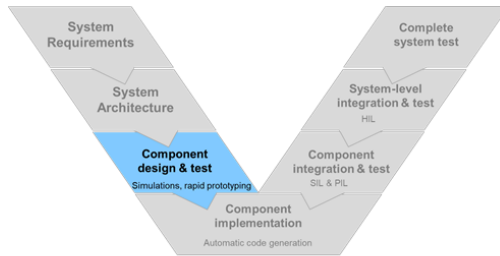
Open Part 1



LSTMs carry a memory cell (state) throughout

AI modeling demo: Creating and training the LSTM model

Use LSTM networks to capture time dependencies in time-series data



AI Modeling



Model design and tuning



Hardware accelerated training



Interoperability

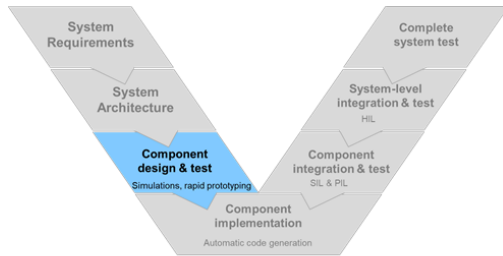
The screenshot shows the MATLAB Live Editor interface. The main window displays the 'Create network' section, where a 'deepNetworkDesigner' object is used to create an LSTM network. The code in the editor is as follows:

```
40 % num of dimmensions of feature
41 numFeatures = 4;
42 numResponses = 1;
43 % num of nodes of
44 numHiddenUnits = 50;
45
46 % Create Network
47 layers = [ ...
48     sequenceInputLayer(numFeatures, "Name", "lstm", "OutputMode", "sequence")
49     lstmLayer(numHiddenUnits, "Name", "lstm", "OutputMode", "sequence")
50     fullyConnectedLayer(numHiddenUnits, "Name", "fc")
51     fullyConnectedLayer(numResponses, "Name", "fc2")
52     regressionLayer("Name", "regressionoutput")];
53 lgraph = layerGraph(layers);
54
55 % setup training options
56 maxEpochs = 100;
```

A callout box highlights the text: "Create a network using 'Deep Network Designer'". The workspace on the left shows variables like 'std_transie...', 'tdataLength', 'testdataidx', 'ThrPosPct', 'training', 'WgAreaPct', 'X_transient', 'Y_transient', and 'Ymu'.

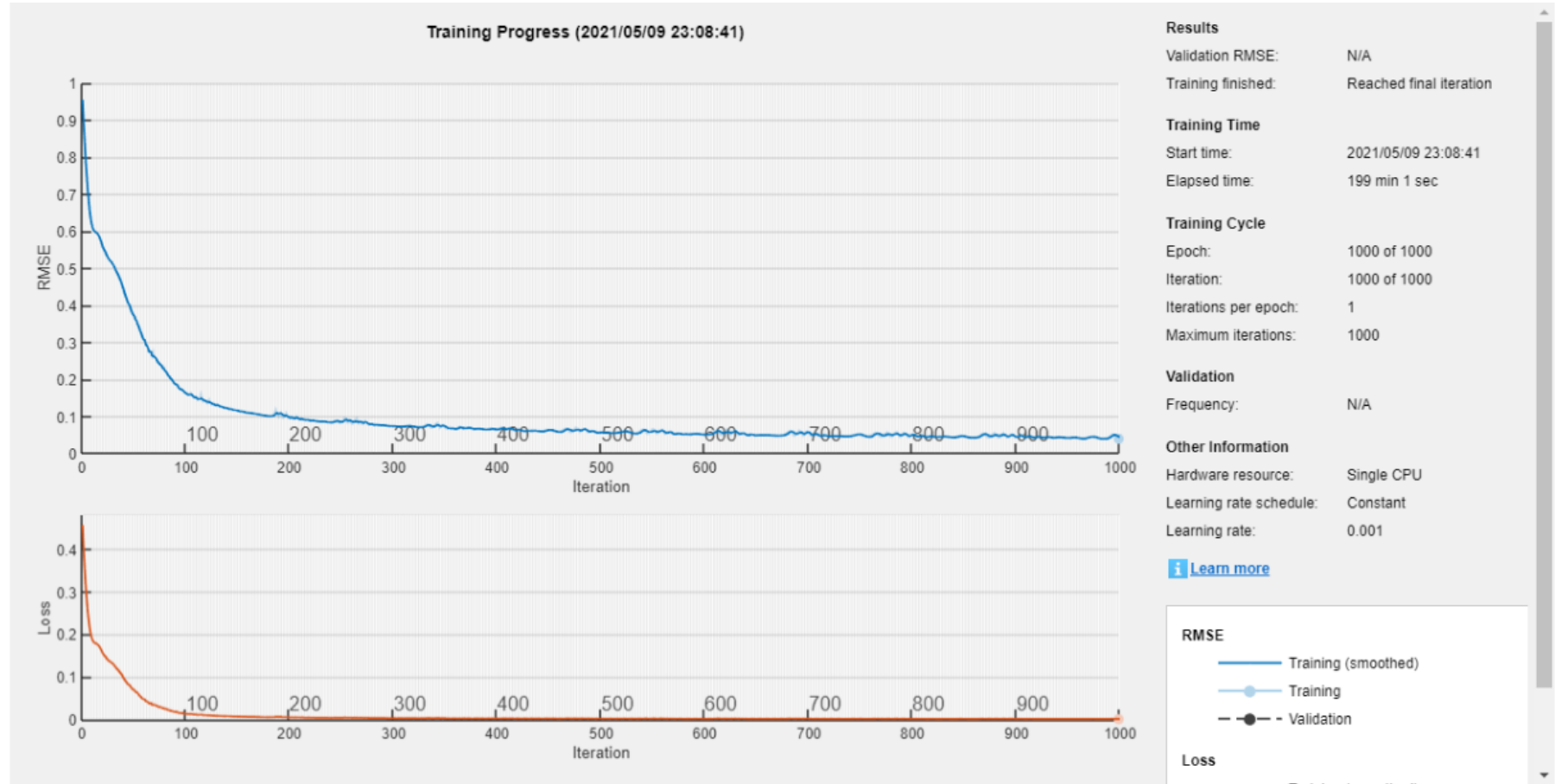
AI modeling demo: Creating and training the LSTM model

Training results



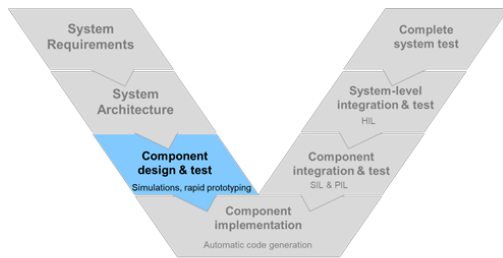
AI Modeling

- Model design and tuning
- Hardware accelerated training
- Interoperability



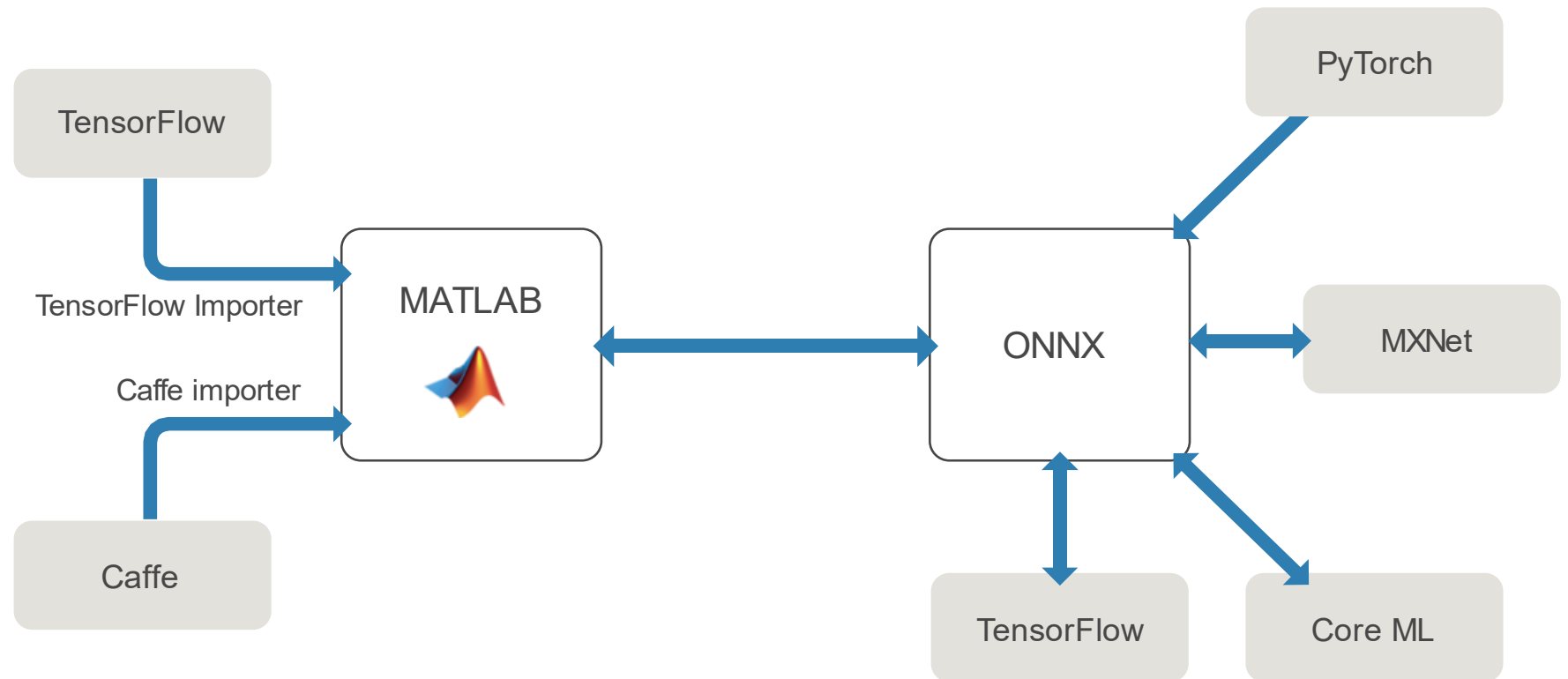
MATLAB interoperates with other frameworks

Framework Interoperability bridges the gap between data science, engineering and production



AI Modeling

- Model design and tuning
- Hardware accelerated training
- Interoperability




AI-driven system design

Data Preparation


 Data cleansing and preparation

 Human insight

 Simulation-generated data

AI Modeling

 Model design and tuning

 Hardware accelerated training

 Interoperability

Simulation & Test

 Integration with complex systems

 System simulation

 System verification and validation

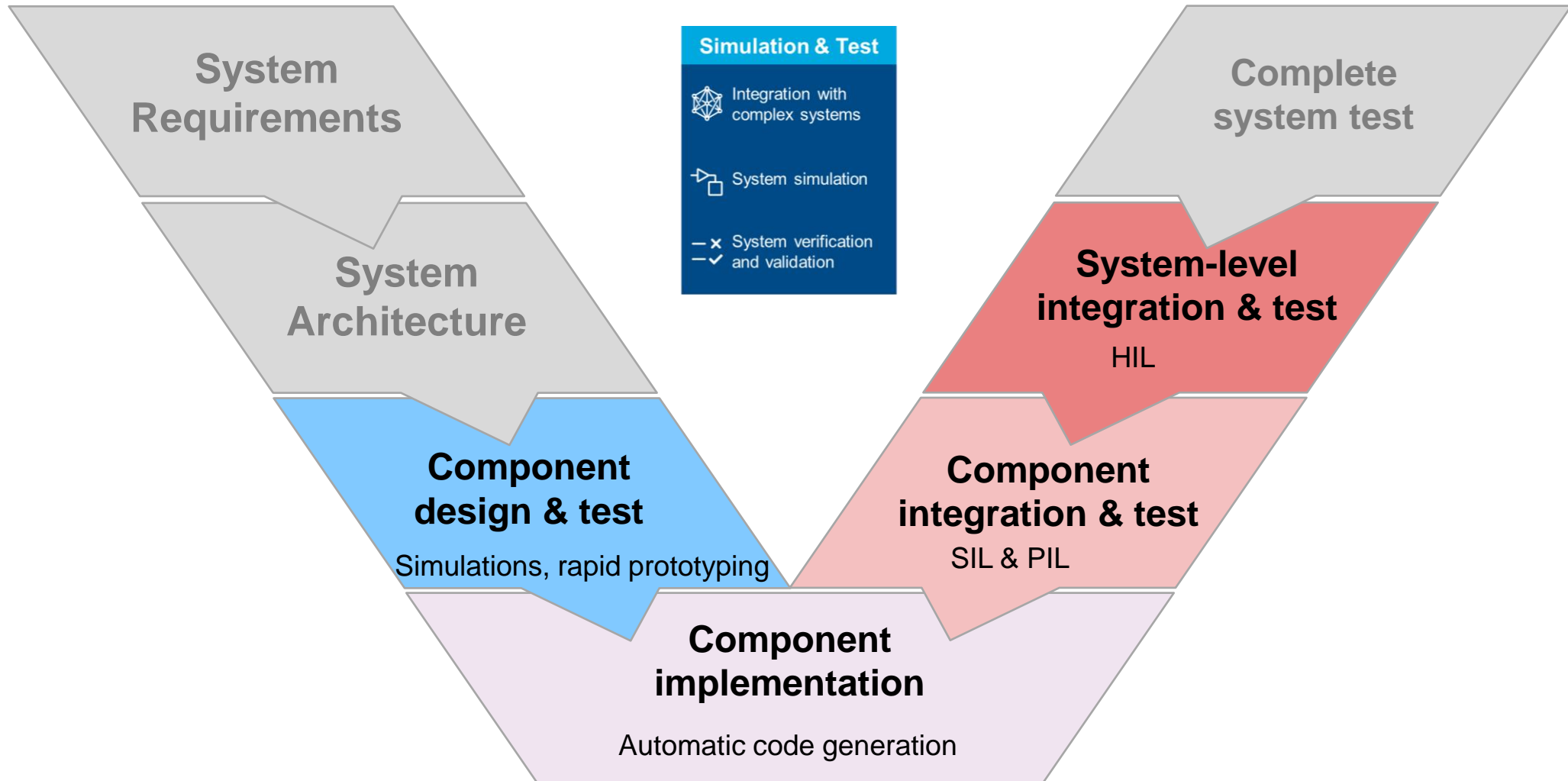
Deployment

 Embedded devices

 Enterprise systems

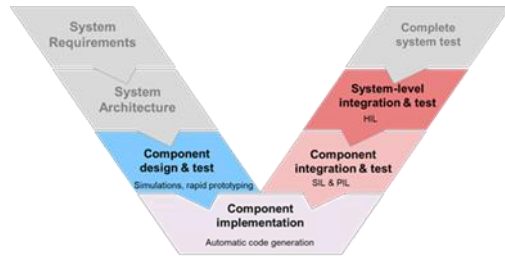
 Edge, cloud, desktop

AI system development with Model-Based Design



Use deep learning within *entire system models* in Simulink


Integrate your deep learning model with your other algorithms for system-level simulation and test

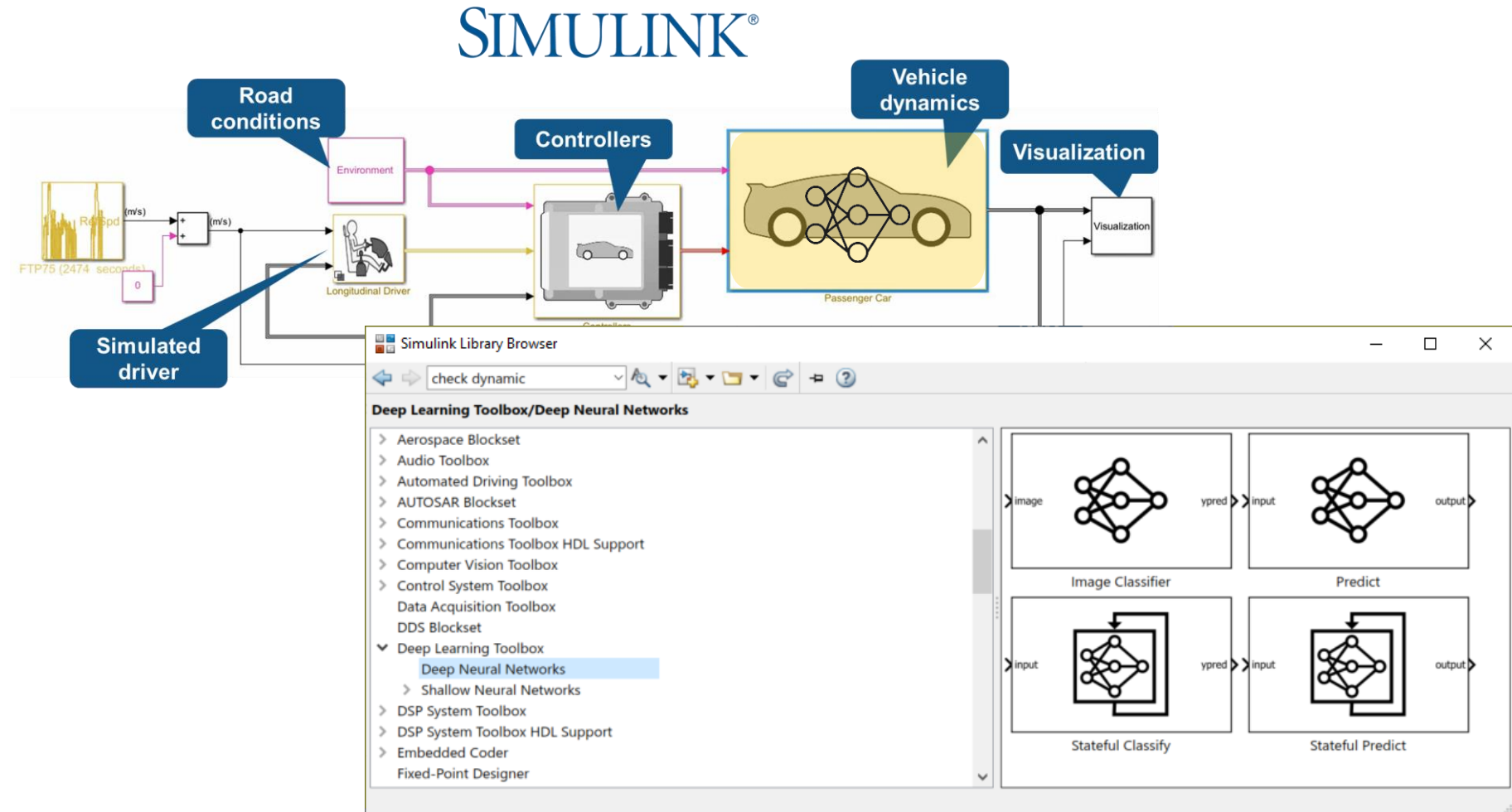


Simulation & Test

 Integration with complex systems

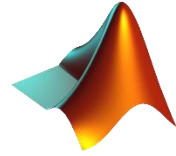
 System simulation

 System verification and validation



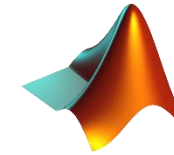
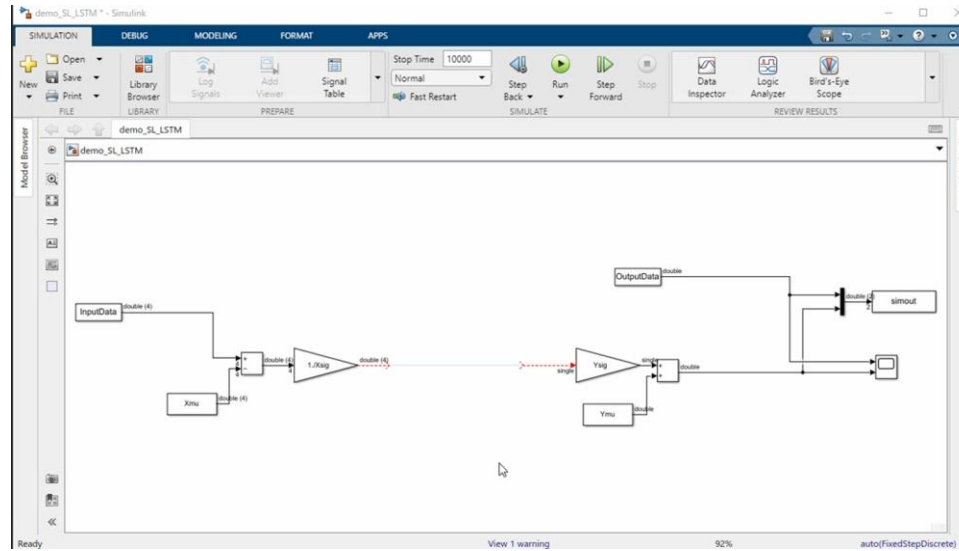
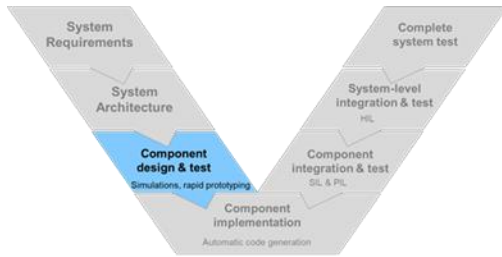
Use Deep Learning Toolbox block library to bring trained deep learning models into Simulink

Simulation demo



Open Part 2

Integration of trained AI model into Simulink

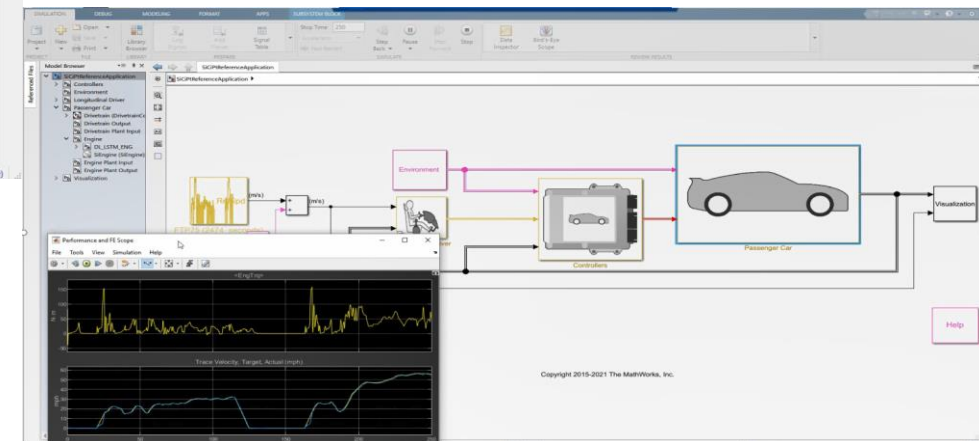


Open Part 3

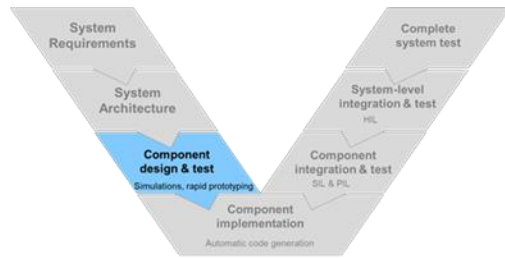
System-level simulation

Simulation & Test

- Integration with complex systems
- System simulation
- System verification and validation



Simulation demo: Integration of trained AI model into Simulink

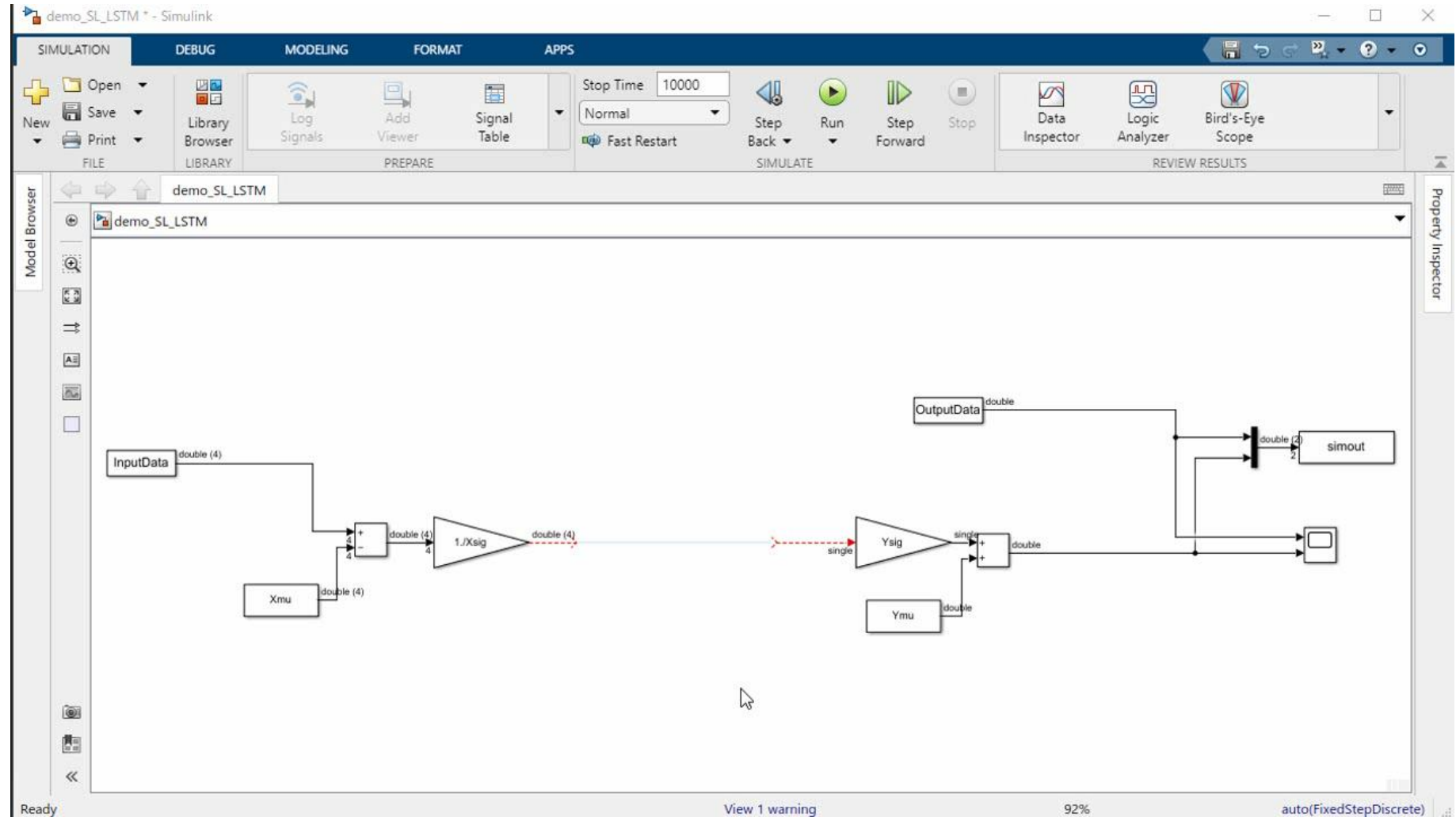


Simulation & Test

 Integration with complex systems

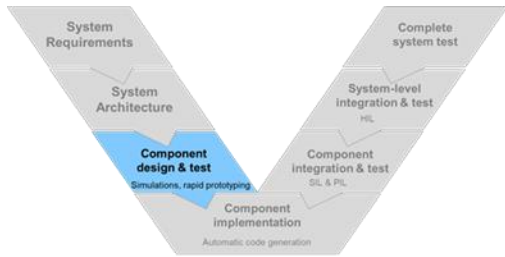
 System simulation

 System verification and validation



A GPU is not required here because the neural network is not very deep


Simulation demo: System-level simulation



Simulation & Test

 Integration with complex systems

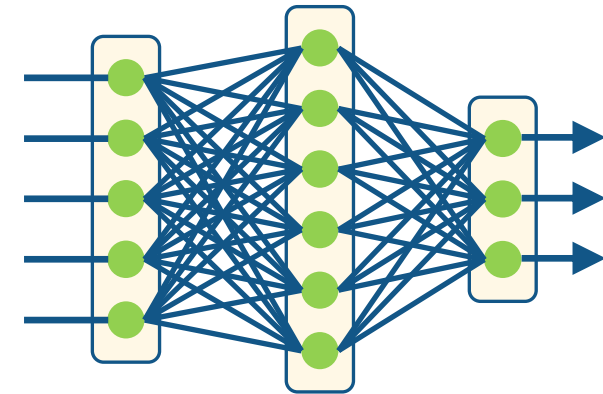
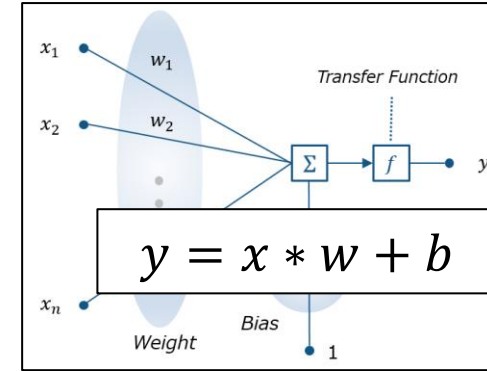
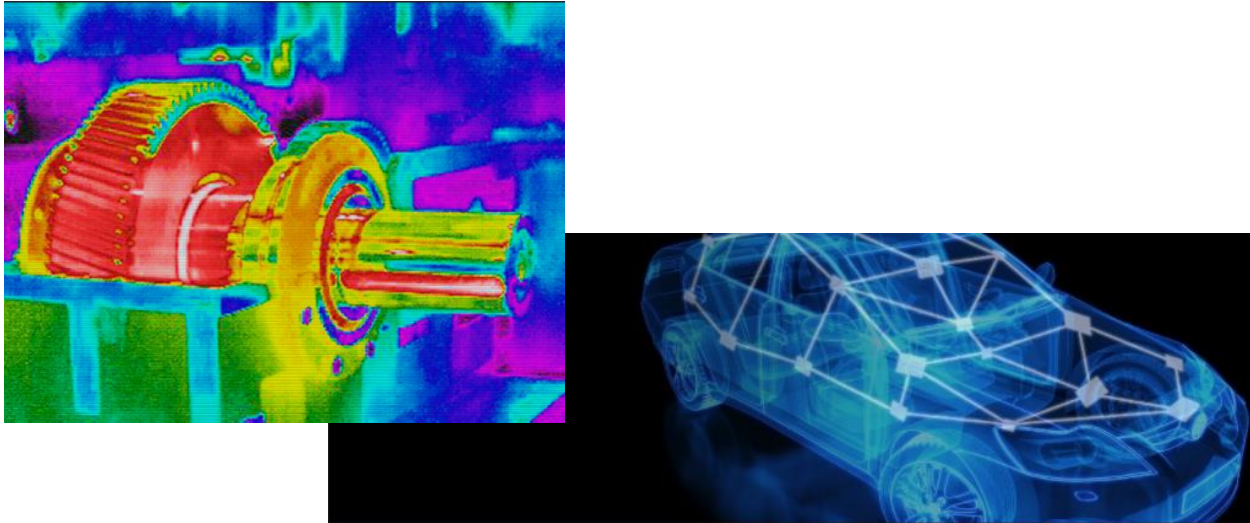
 System simulation

 System verification and validation

Copyright 2015-2021 The MathWorks, Inc.

Speeding up with ROM (Reduced Order Model)

- Detailed simulation with 3D-CAE tool (Large calculation load per step)



- Deep Learning model
 - ✓ Reduced order
 - ✓ Libraries for inference
 - ✓ Parallelization on GPU


Dimension compression and simulation speed-up using deep learning models

AI-driven system design

Data Preparation


 Data cleansing and preparation

 Human insight

 Simulation-generated data

AI Modeling

 Model design and tuning


 Hardware accelerated training

 Interoperability

Simulation & Test

 Integration with complex systems

 System simulation

 System verification and validation

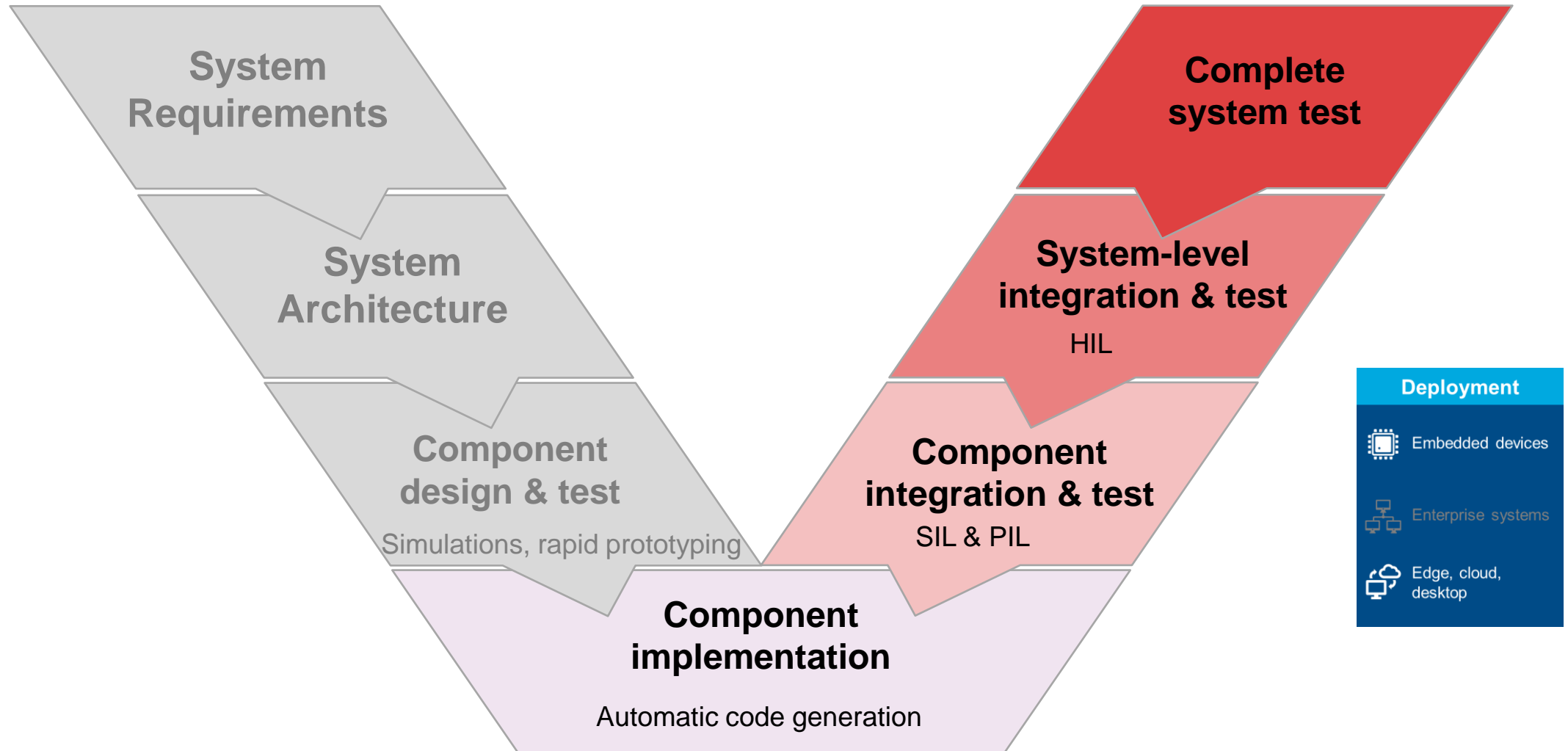
Deployment

 Embedded devices

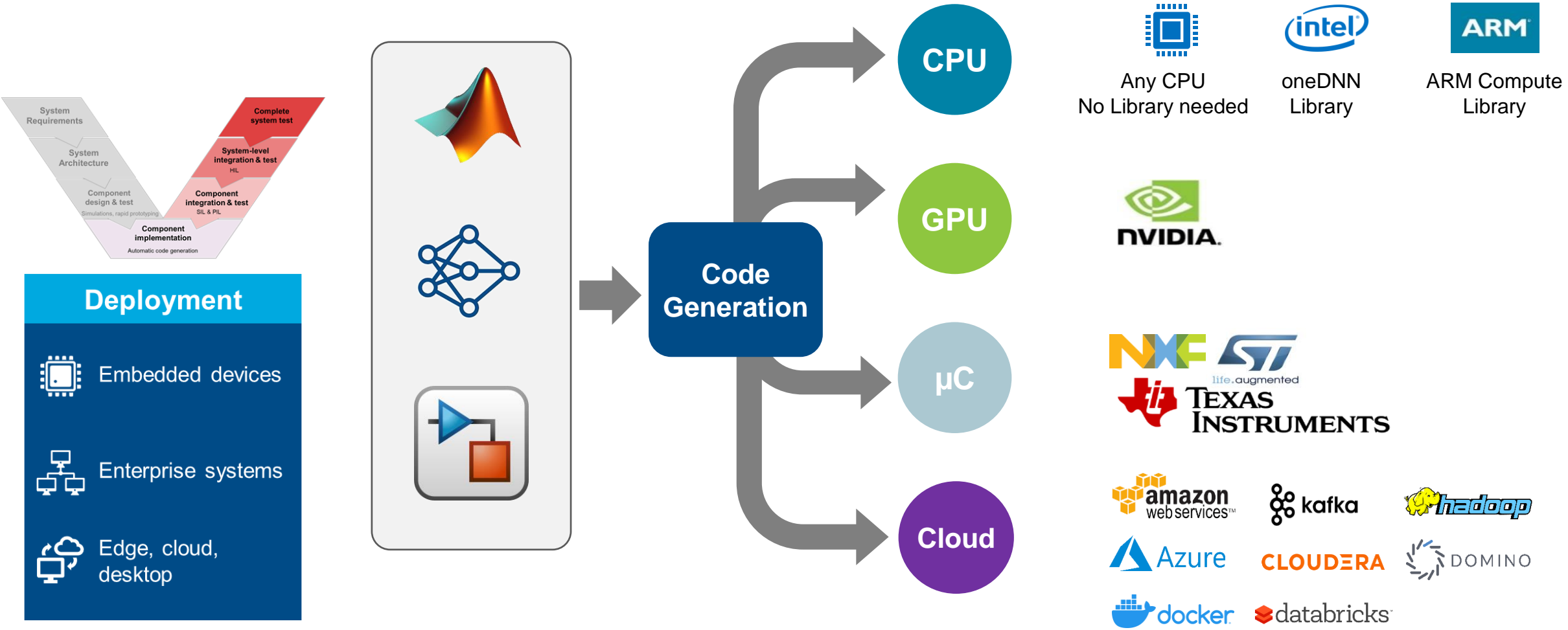
 Enterprise systems

 Edge, cloud, desktop

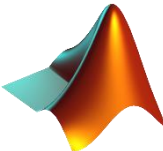
AI system development with Model-Based Design



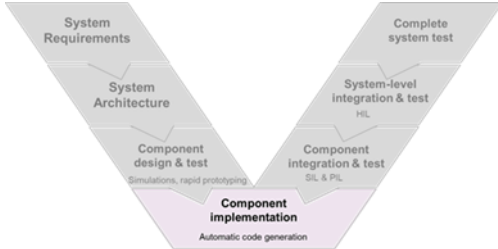
Deploy to target with zero coding errors



Deployment demo: Automatic C code generation



Open Part 4



Deployment

- Embedded devices
- Enterprise systems
- Edge, cloud, desktop

Generate C code

demo_SL_LSTM * - Simulink

Code Generation Report

Contents

- Summary
- Subsystem Report
- Code Interface Report

Generated Code

- [-] Model files
 - demo_SL_LSTM.c
 - demo_SL_LSTM.h
 - demo_SL_LSTM_private.h
 - demo_SL_LSTM_types.h
- [-] Data files
 - demo_SL_LSTM_data.c
- [+] Utility files (9)
- [+] Interface files (1)
- [+] Other files (1)

```
11  /* Note: GRT includes extra infrastructure and instrumentation for prototyping
12  * Embedded hardware selection: Intel->>86-64 (Windows64)
13  * Code generation objective: Debugging
14  * Validation result: Not run
15  */
16
17  #include "demo_SL_LSTM.h"
18  #include "demo_SL_LSTM_private.h"
19
20  /* Block signals (default storage) */
21  B_demo_SL_LSTM_T demo_SL_LSTM_B;
22
23  /* Block states (default storage) */
24  DW_demo_SL_LSTM_T demo_SL_LSTM_DW;
25
26  /* Real-time model */
27  static RT_MODEL_demo_SL_LSTM_T demo_SL_LSTM_M;
28  RT_MODEL_demo_SL_LSTM_T *const demo_SL_LSTM_M = &demo_SL_LSTM_M;
29
30  /* Forward declaration for local functions */
31  static void DeepLearningNetwork_setSizeDepe(c_coder_ctarget_DeepLearningN_T *obj);
32  static void demo_SL_LSTM_LSTMlayer_predict(const real32_T X[4], const real32_T
33  cellState[50], const real32_T hiddenState[50], real32_T Y[50], real32_T CS[50],
34  real32_T HS[50]);
35  static real32_T DeepLearningNetwork_predictAndU(c_coder_ctarget_DeepLearningN_T *
36  obj, const real_T indata[4]);
37
38  /* Function for MATLAB Function: '<S1>/MLFB' */
39  static void DeepLearningNetwork_setSizeDepe(c_coder_ctarget_DeepLearningN_T *obj)
40  {
```

Model Browser

Code M... Ready

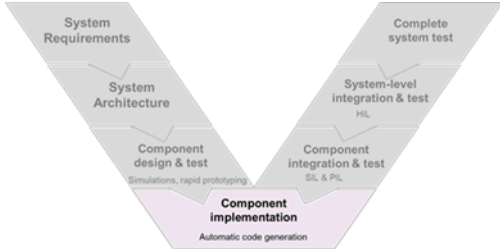
Property Inspector

simout

Finish

auto(FixedStepDiscrete)

Deployment demo: Automatic C code generation

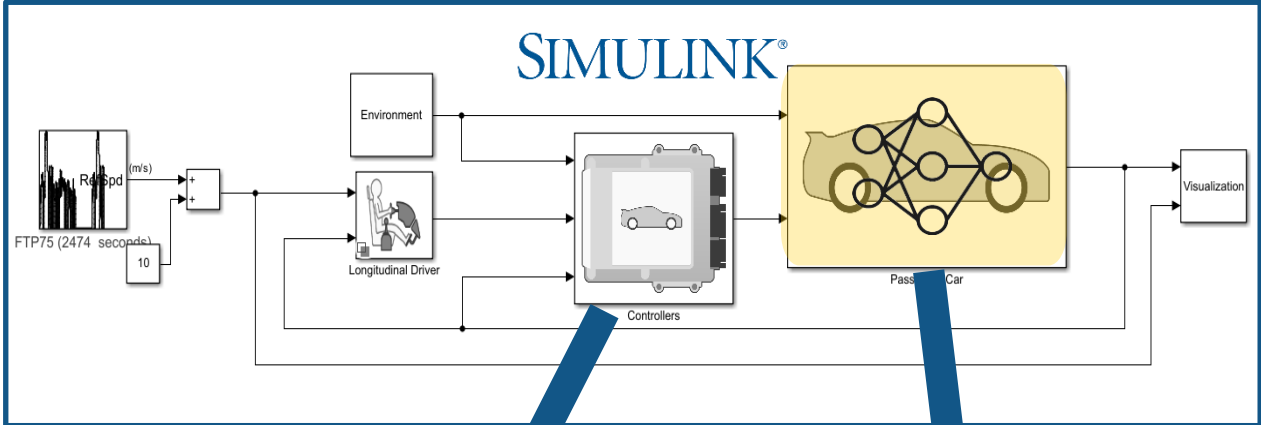


Deployment

- Embedded devices
- Enterprise systems
- Edge, cloud, desktop

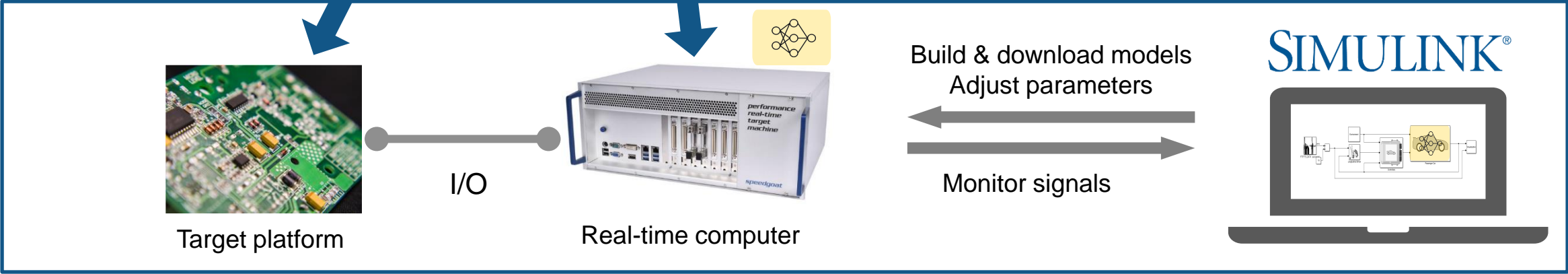
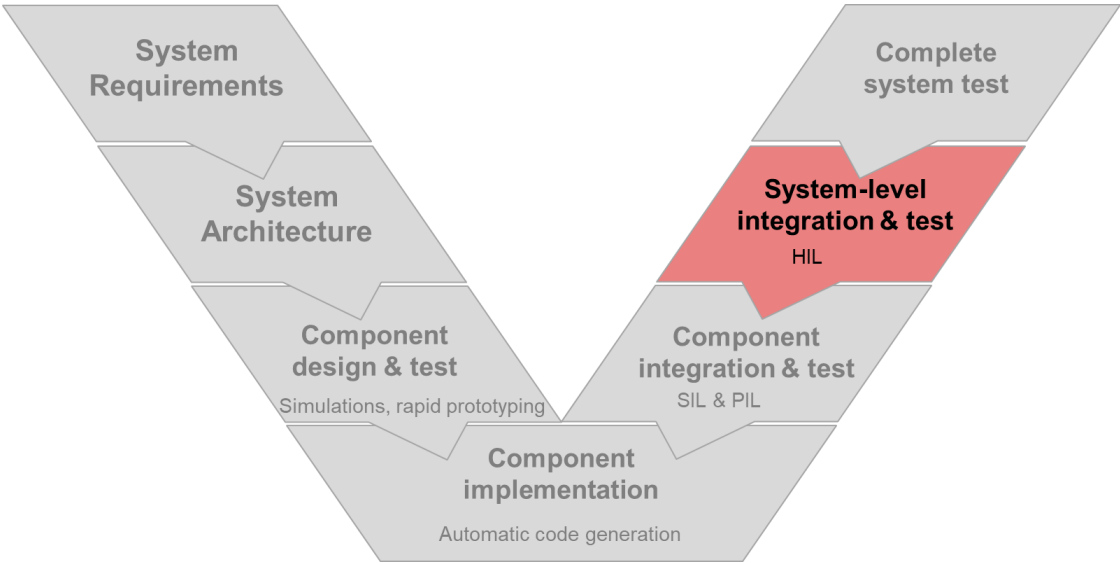
The screenshot shows the Simulink C Code generation interface for a model named 'demo_SL_LSTM'. The 'C CODE' tab is active, displaying the 'Code for demo_SL_LSTM' section. A prominent 'Generate C code' button is visible in the top right. The main workspace shows a Simulink block diagram with inputs 'InputData' (double (4)) and 'Xmu' (double (4)), and outputs 'OutputData' (double) and 'simout'. The diagram includes a 'Sequenceinput' block with an LSTM neural network icon, a gain block '1./Xsig', and a 'Ysig' block. The status bar at the bottom indicates 'Code Mappings - C', 'Ready', 'View 2 warnings', '94%', and 'auto(FixedStepDiscrete)'.

Deployment demo: Hardware-in-the-loop simulation

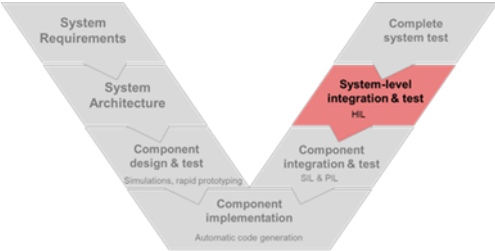


Code generation from algorithm

Code generation from plant model



Deployment demo: Hardware-in-the-loop simulation



Deployment

- Embedded devices
- Enterprise systems
- Edge, cloud, desktop

The screenshot shows the Simulink environment for a project named 'demo_SL_LSTM'. The 'C CODE' tab is active, displaying a 'Generate C code' button. The main workspace contains a Simulink block diagram. It starts with an 'InputData' block (double (4)) and an 'Xmu' block (double (4)) which are summed and multiplied by 1./Xsig. The result goes into a 'Sequenceinput' block (double (4)) which feeds into an LSTM block. The LSTM block's 'regressionoutput' (single) is summed with 'Ymu' (double) and then multiplied by 'Ysig' (single). The final output is 'OutputData' (double), which is also summed with another 'double (2)' signal before being sent to a 'simout' block. The status bar at the bottom shows 'Code Mappings - C', 'Ready', 'View 2 warnings', '94%', and 'auto(FixedStepDiscrete)'.

Conclusion

Replacing a first-principles engine model with an AI-based surrogate

First principles models and data-driven models can co-exist

First-principles models

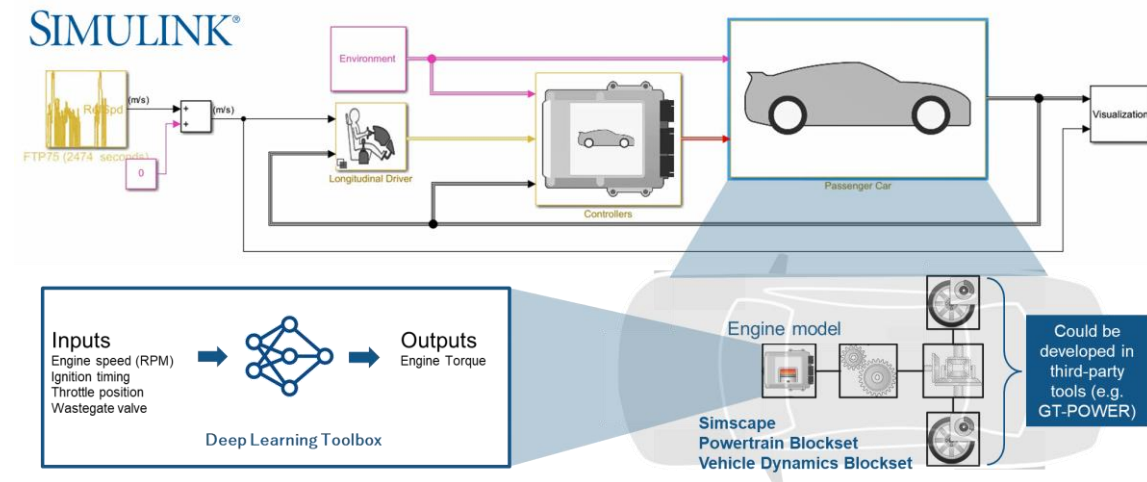
Data-driven models

White-box

Gray-box

Black-box

- Preprocessed synthetic data generated from Simulink
- Trained a deep learning-based LSTM model to replace part of the complex dynamics of a vehicle engine
- Integrated trained network into Simulink for system-level simulation together with first-principles components
- Generated C code and performed HIL tests





MathWorks: helping engineers & scientists build Deep Learning solutions



The Platform

MATLAB, Simulink, and over 100 add-on products for specialized applications



Your People

Helping you build an agile workforce today and preparing tomorrow's engineers



Our Expertise

From onboarding and implementation to solving advanced engineering challenges



감사합니다

송완빈 과장

Application Engineer @ MathWorks

wsong@mathworks.com

