MATLAB/Simulink Training for Smart Model Car

류성연 차장
Senior Application Engineer
MathWorks Korea

Agenda

- 모델 기반 설계 개요
- 시뮬링크를 이용한 개발 환경 구축
- 자율 주행 개발 방안
- 모델링 및 코드 생성 실습

Headquarters

Natick, MA USA

North America

United States

Europe

France
Germany
Ireland
Italy
Netherlands
Spain
Sweden
Switzerland
UK

Asia-Pacific

Australia
China
India
Japan
Korea

3 million+ users

in more than 180 countries

4000+ staff

in 31 offices around the world

$900+ million

in 2017 revenues with 60% from outside the US

Privately held

and profitable every year

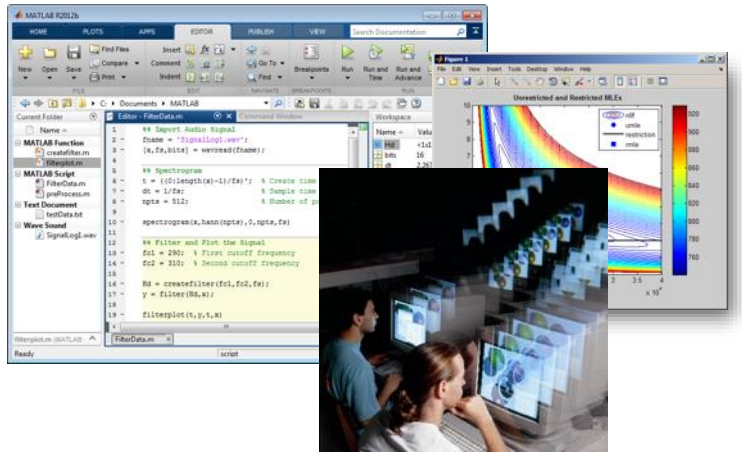
MathWorks

Accelerating the Pace of Engineering and Science

MATLAB®

The leading environment for technical computing

The industry-standard, high-level programming language for algorithm development, numeric computation, data analysis and visualization



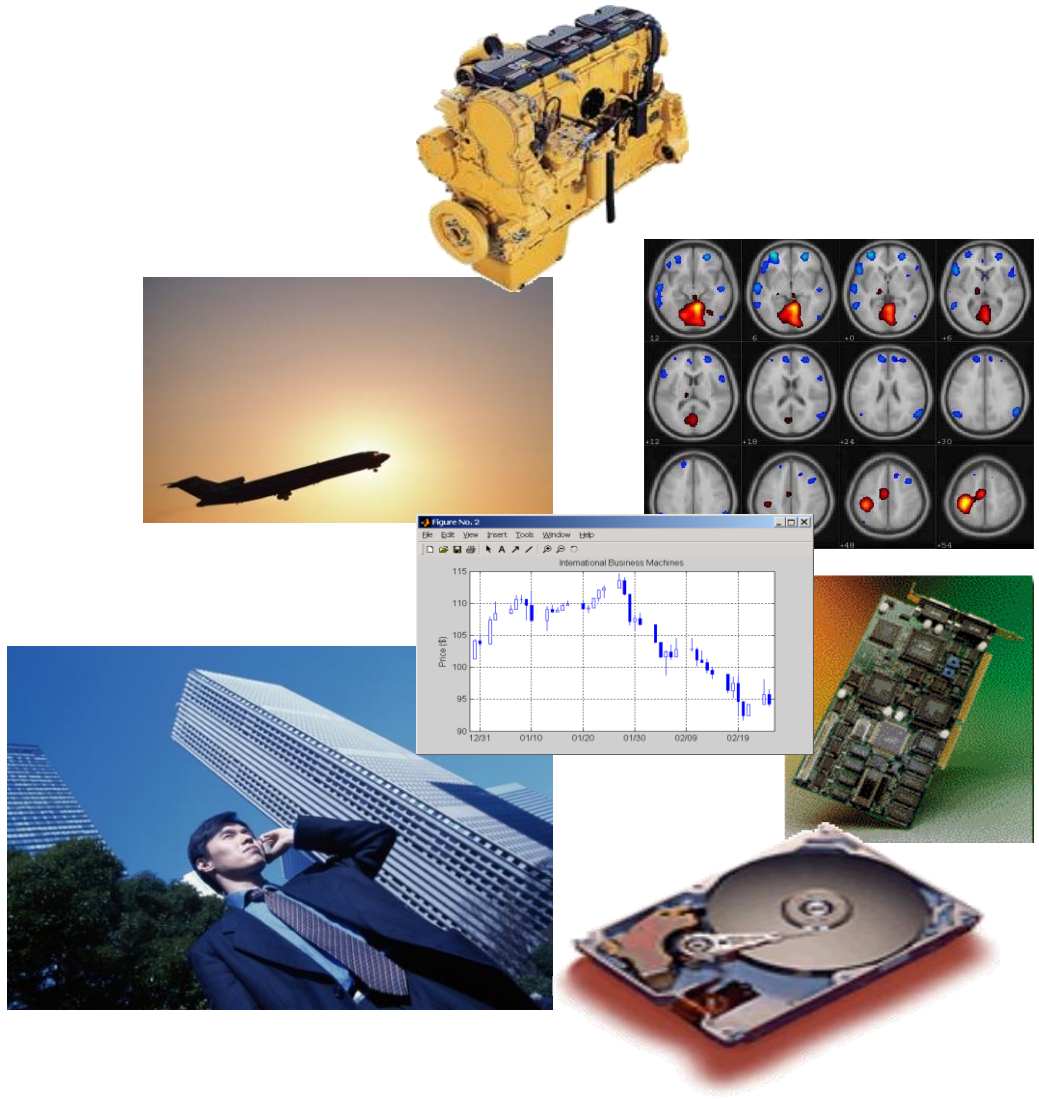
SIMULINK®

The leading environment for modeling, simulating, and implementing dynamic and embedded systems

Applications in controls, signal processing, physical modeling, robotics, and other system engineering areas

Key Industries

- Aerospace and defense
- Automotive
- Biotech and pharmaceutical
- Communications
- Education
- Electronics and semiconductors
- Energy production
- Financial services
- Industrial automation and machinery
- Medical devices
- Software
- Internet



MathWorks Product Overview

Event-Based Modeling
- Stateflow
- SimEvents

Physical Modeling
- Simscape
- Simscape Multibody
- Simscape Driveline
- Simscape Fluids
- Simscape Electrical

Real-Time Simulation and Testing
- Simulink Real-Time

Verification, Validation and Test
- Simulink Requirements
- Simulink Check
- Simulink Coverage
- Simulink Design Verifier
- Simulink Test
- Polyspace

Simulation Graphics and Reporting
- Simulink 3D animation
- Simulink Report Generator

SIMULINK®

Simulation and Model-Based Design

Parallel Computing
- Parallel Computing Toolbox

Code Generation
- MATLAB Coder
- Simulink Coder
- Embedded Coder
- Fixed-Point Designer
- GPU Coder
- HDL Coder
- PLC Coder

MATLAB®

The Language of Technical Computing

Math, Statistics, and Optimization
- Statistics and Machine Learning Toolbox
- Curve Fitting Toolbox
- Deep Learning Toolbox

Application Deployment
- MATLAB Compiler

Database Access and Reporting
- Database Toolbox
- MATLAB Report Generator

Applications

Control Systems
- Control System Toolbox
- Powertrain Blockset
- Robotics System Toolbox
- Vehicle Dynamics Blockset

Signal Processing and Communications
- Computer Vision System Toolbox

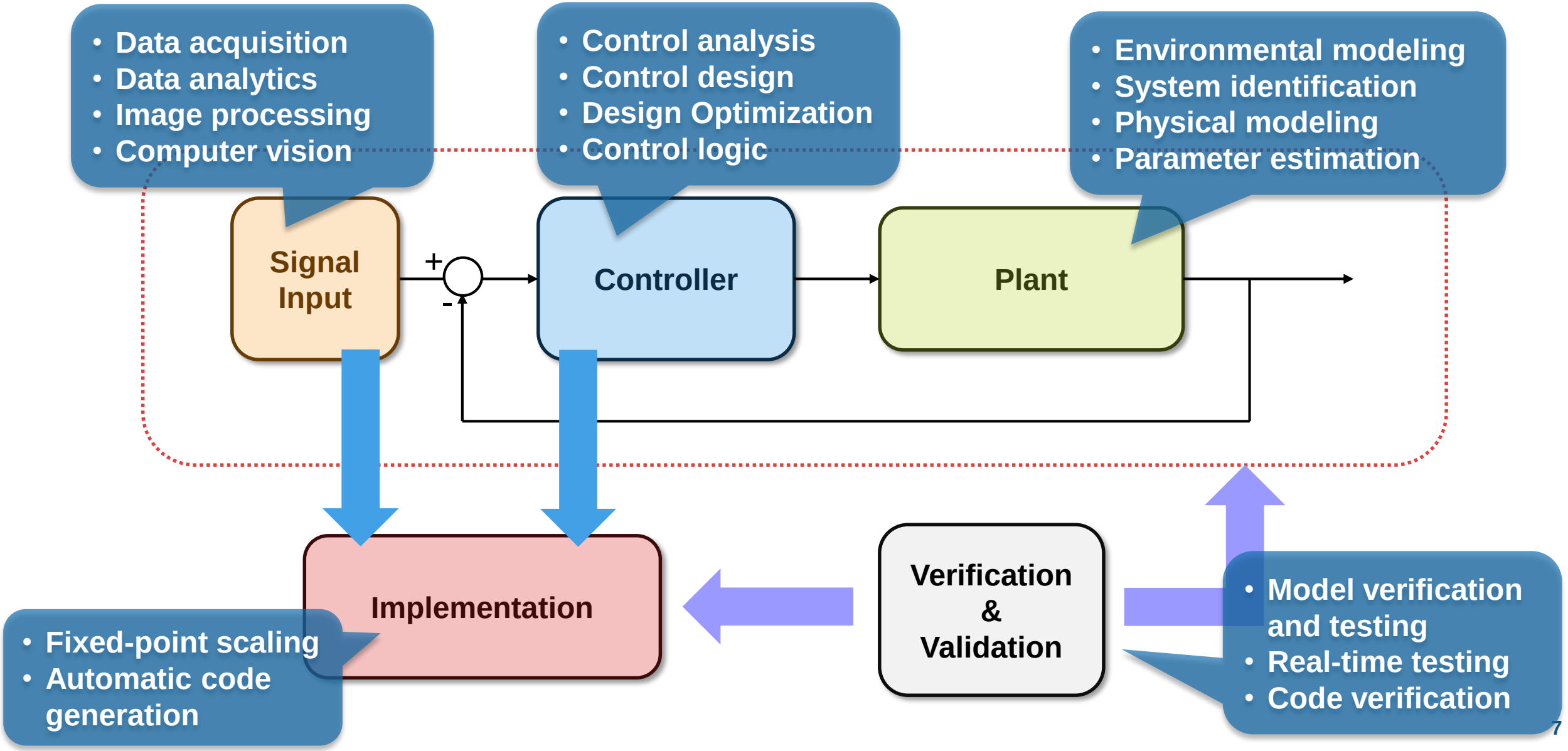
Image Processing and Computer Vision
- Image Processing Toolbox
- Automated Driving System Toolbox

Test and Measurement
- Instrumentation Control Toolbox
- Vehicle Network Toolbox

Computational Finance

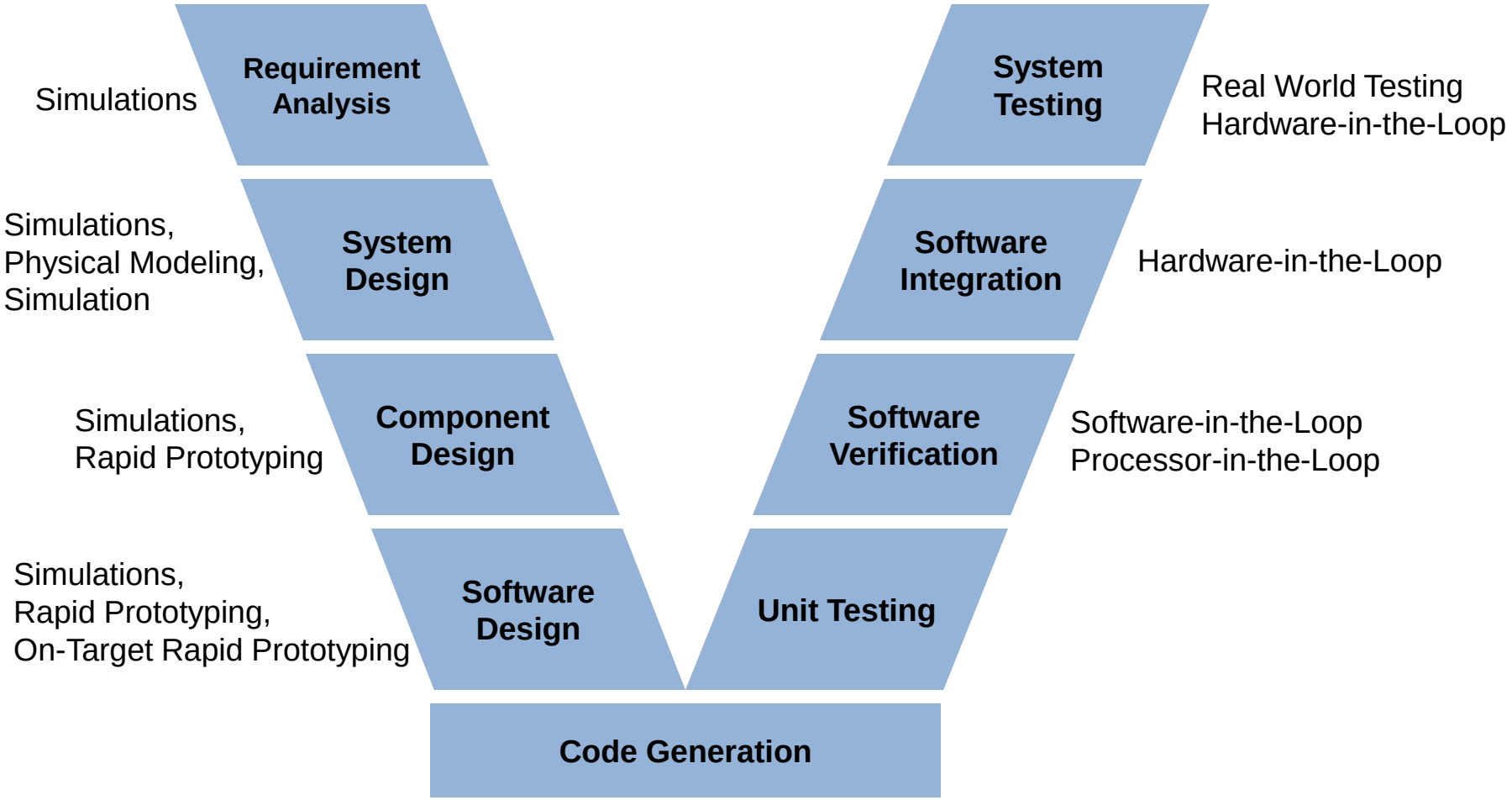
Computational Biology

MATLAB/Simulink for Control System Design

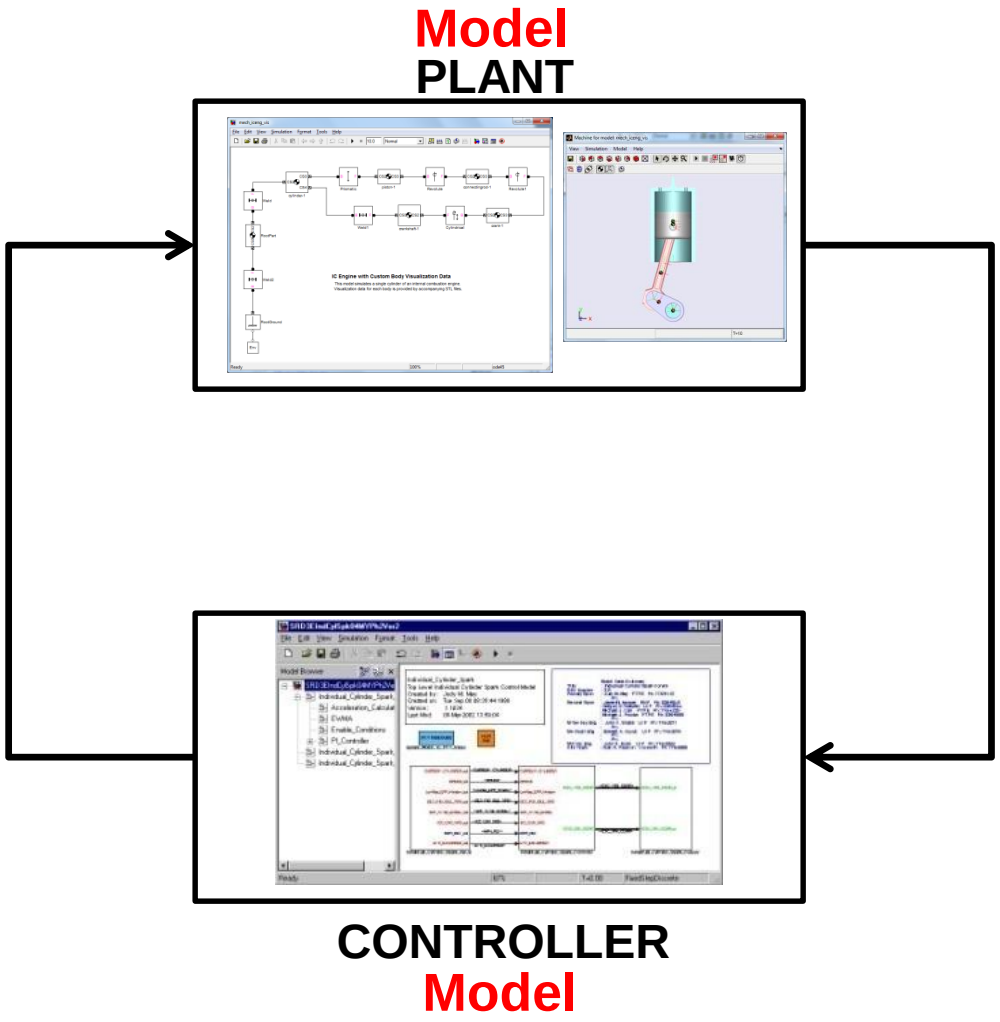


Model-Based Design Overview

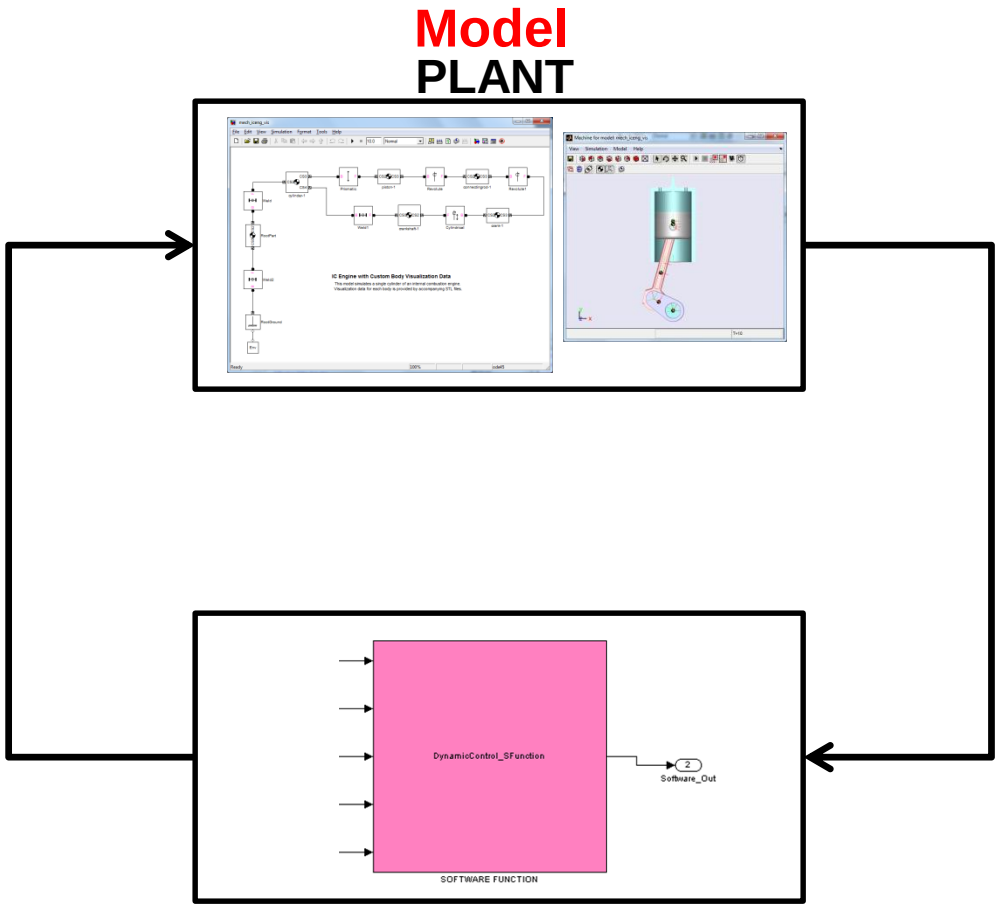
Model Based Design Process



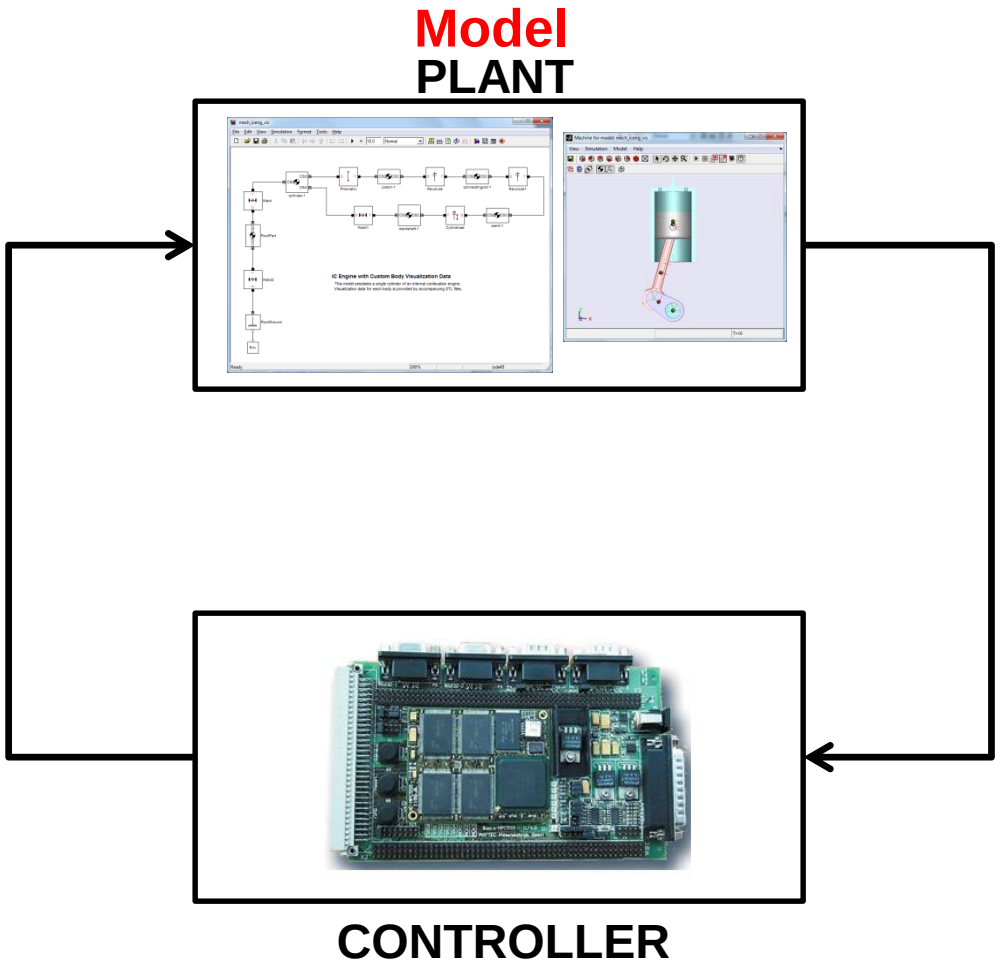
Model Simulation



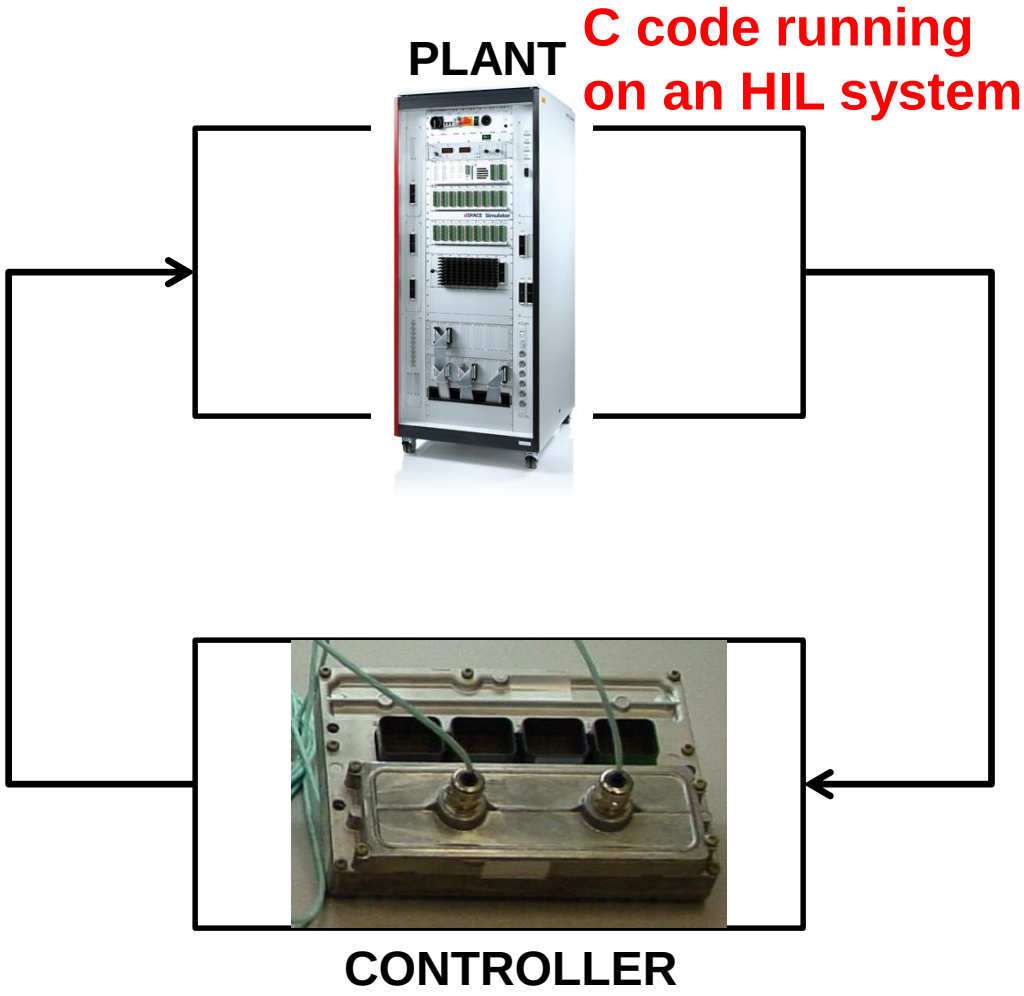
Software-in-the-Loop (SIL)



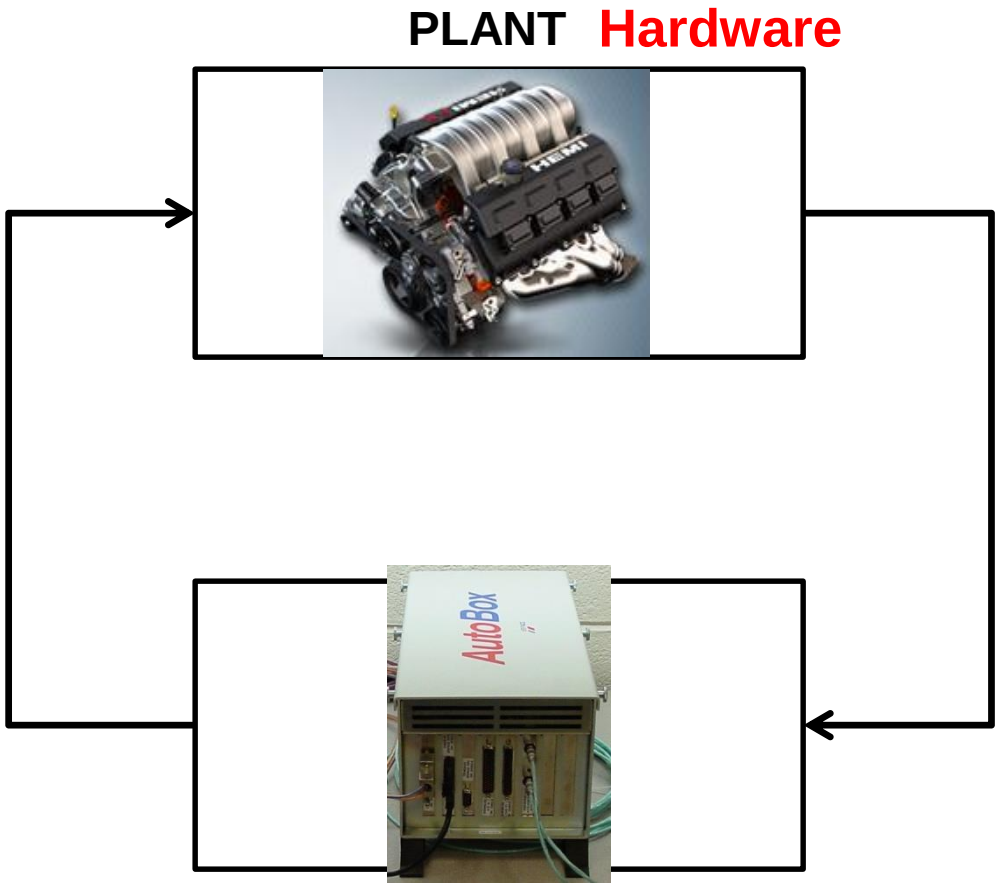
Processor-in-the-Loop (PIL)



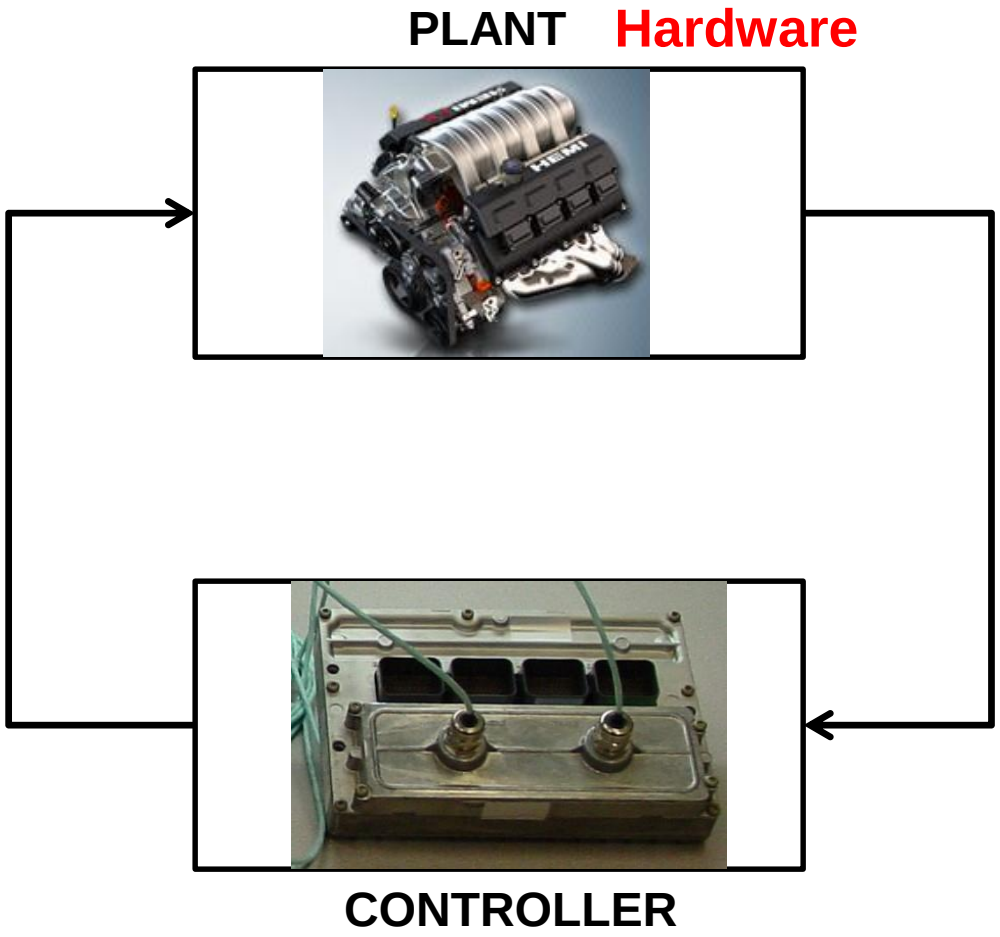
Hardware-in-the-Loop (HIL)



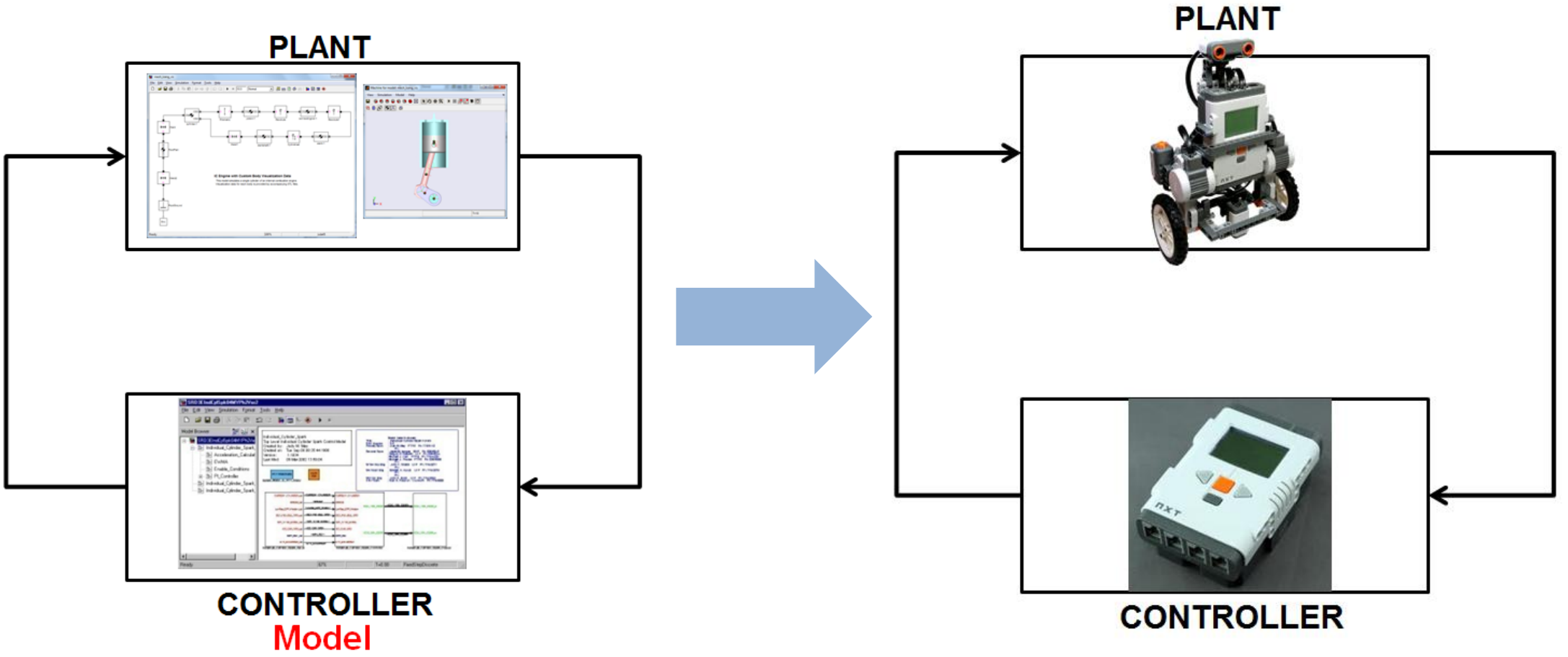
Rapid Prototyping (RP/RCP/RPC)



Deployment


PLANT
Hardware
HEMI
CONTROLLER
C code running on
the target hardware

Customizable Process



Lockheed Martin F-35B

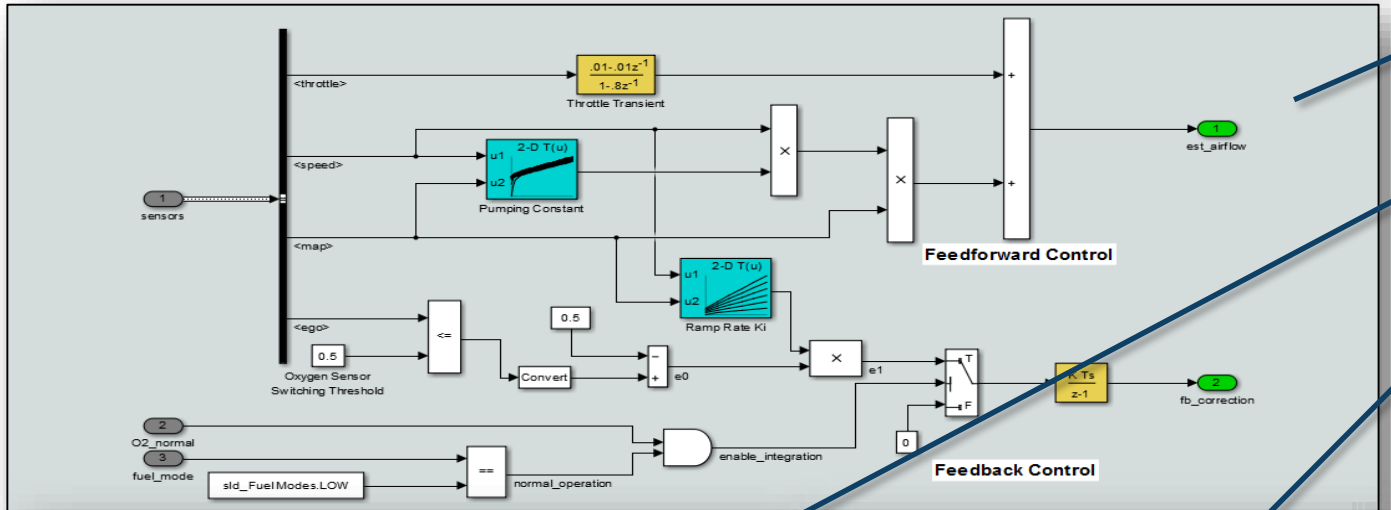
Joint Strike Fighter

Scania

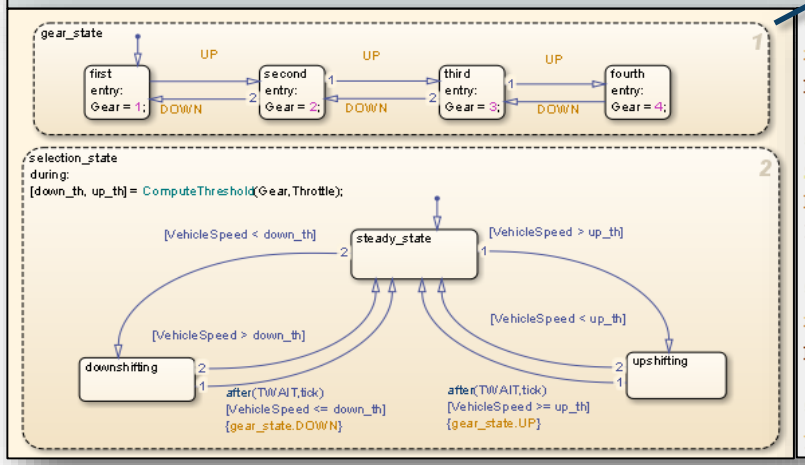
Automatic Emergency Braking

Software Modeling Tools

Simulink



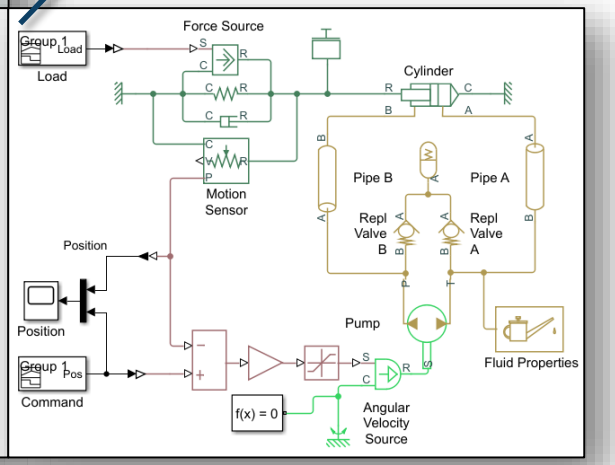
- Block oriented for system design
- State machines for event-based logic
- Textual for concise numerics
- Component based modeling



Stateflow

```
% Predicted state and covariance
x_prd = A * x_est;
p_prd = A * p_est * A' + Q;

% Estimation
C = H * p_prd' * H' + R;
B = H * p_prd';
klm_gain = (S \ B)';

% Estimated state and covariance
x_est = x_prd + klm_gain * (z - H * x_prd);
p_est = p_prd - klm_gain * H * p_prd;

% Compute the estimated measurements
y = H * x_est;
```

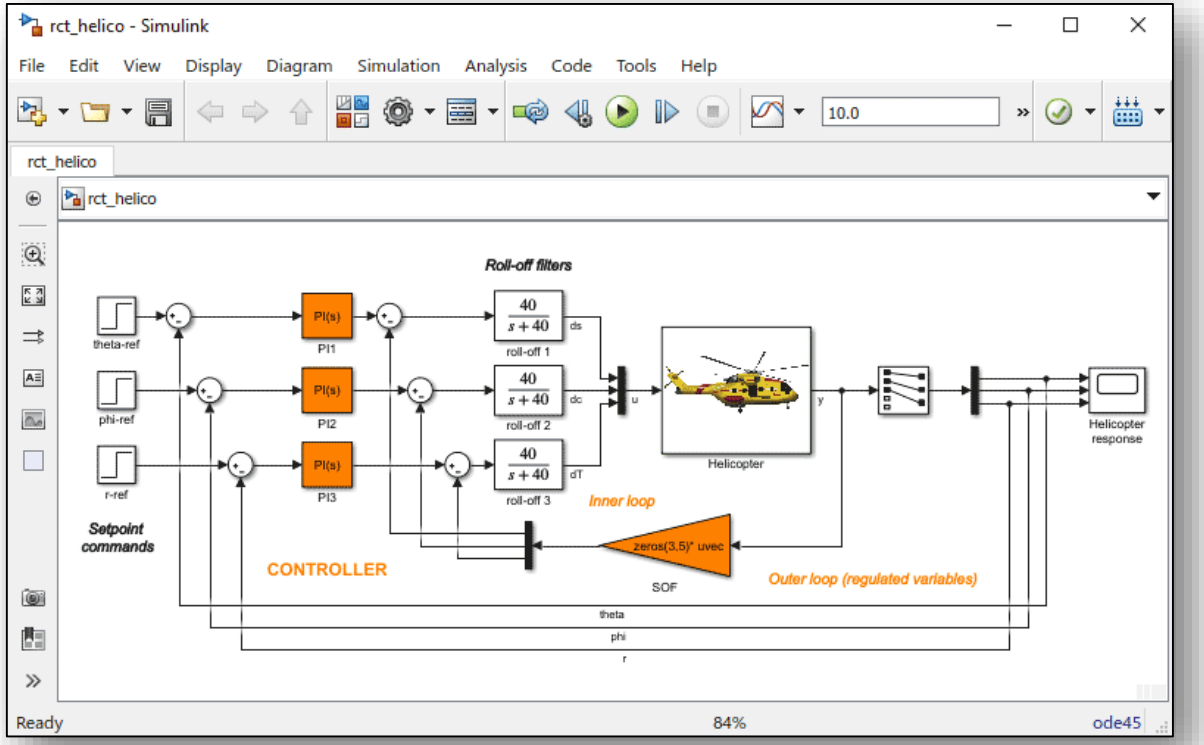
MATLAB

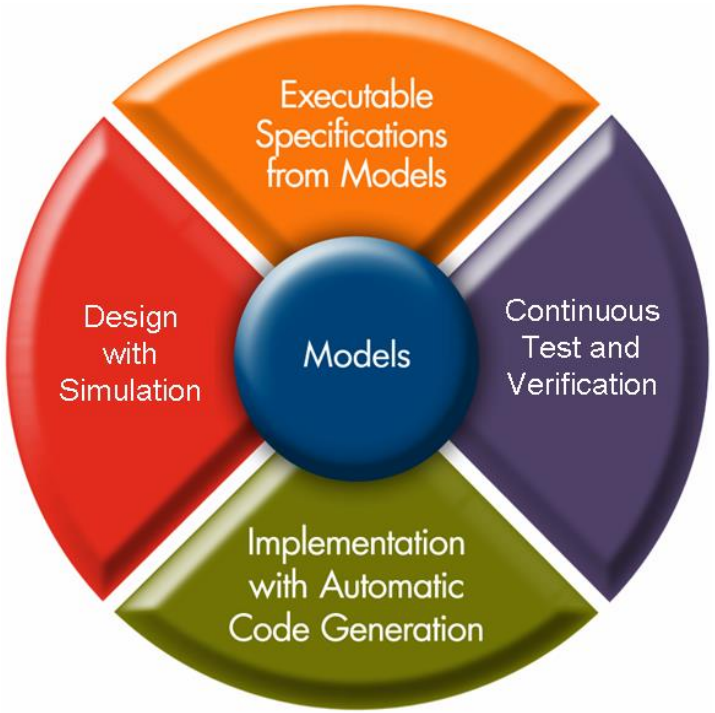


Simscape

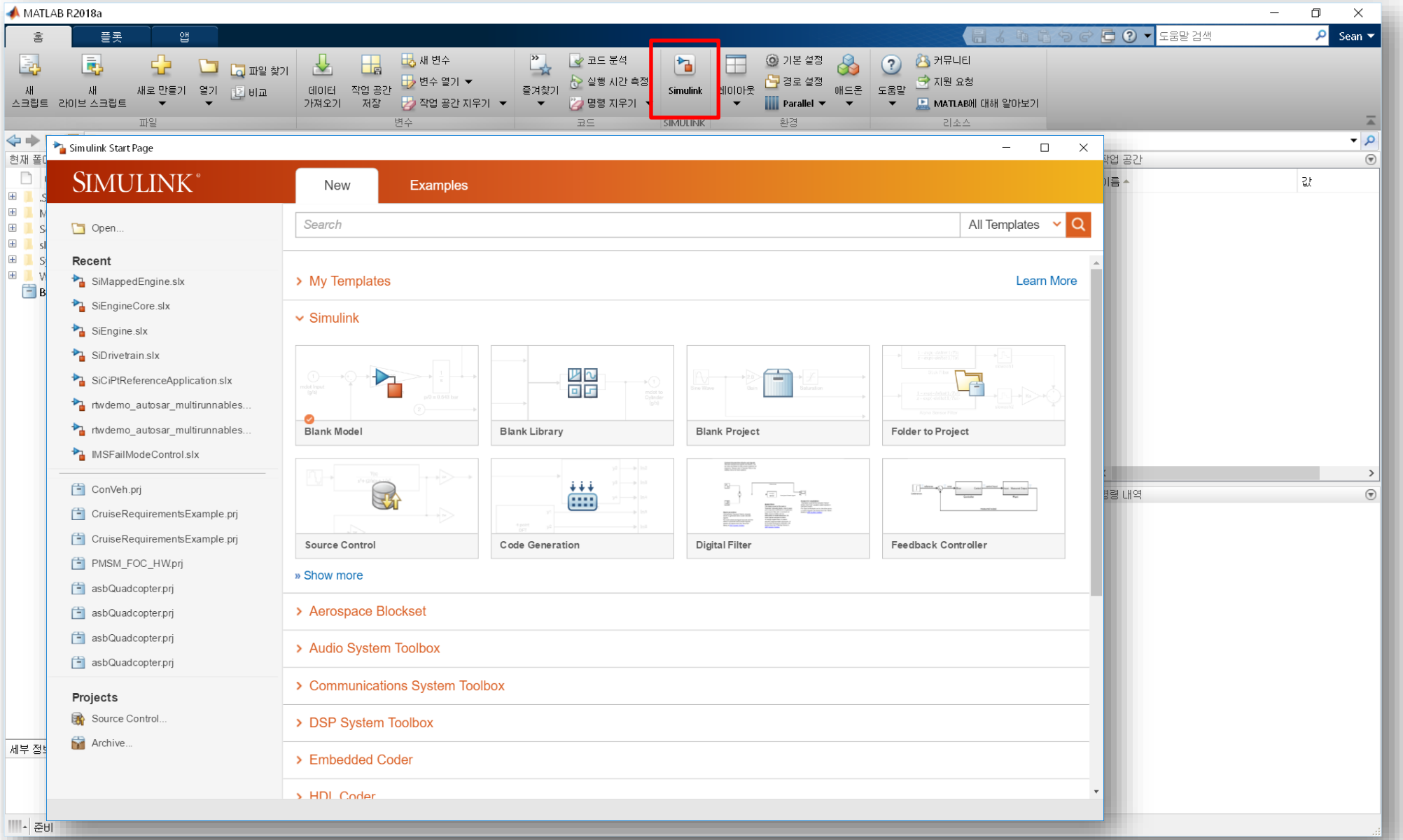
What is Simulink?

- Simulink is **Graphical Modeling**
- Simulink is **Time and Event Domain Simulation**
- Simulink is **Model-Based Design**



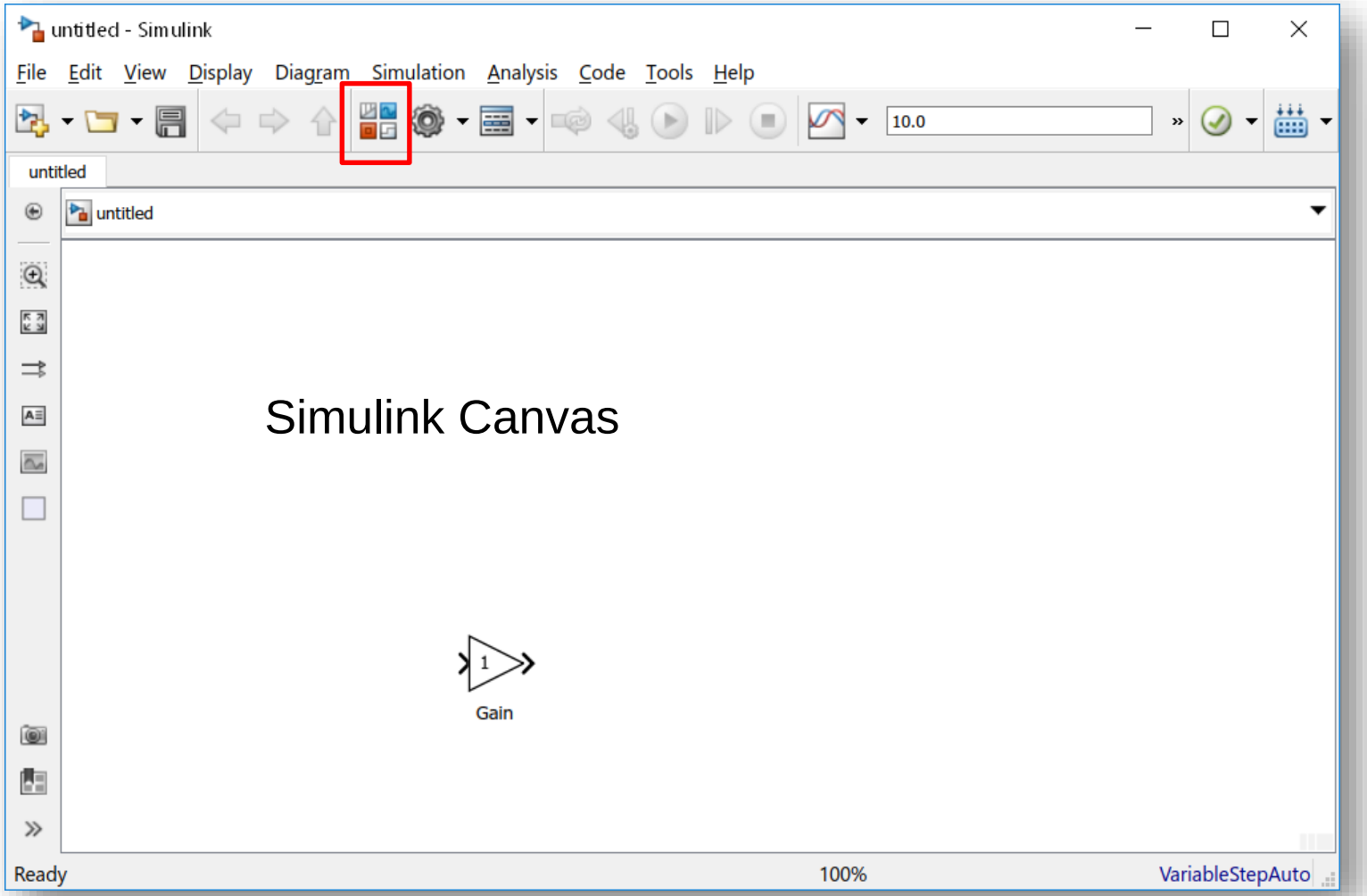


The Simulink® Desktop

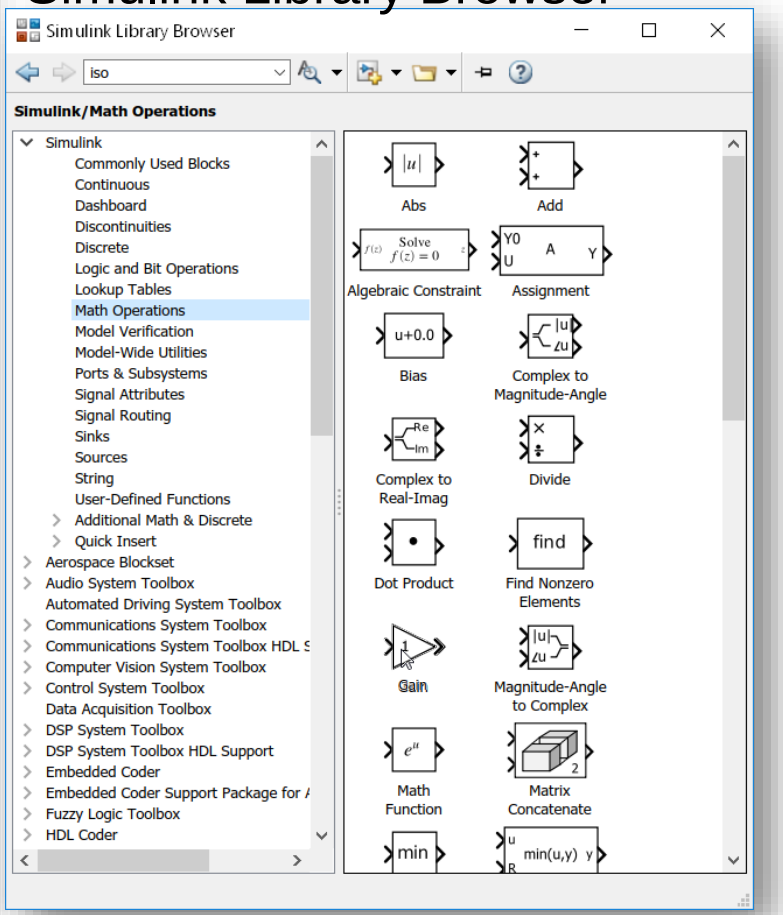


>> simulink

The Simulink® Desktop



Simulink Library Browser

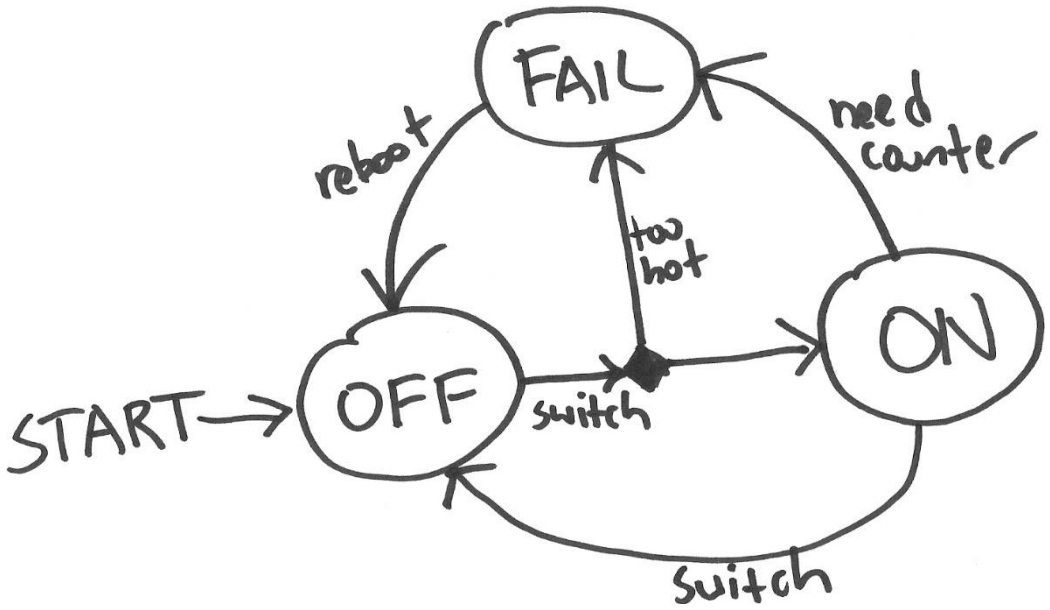


>> slLibraryBrowser

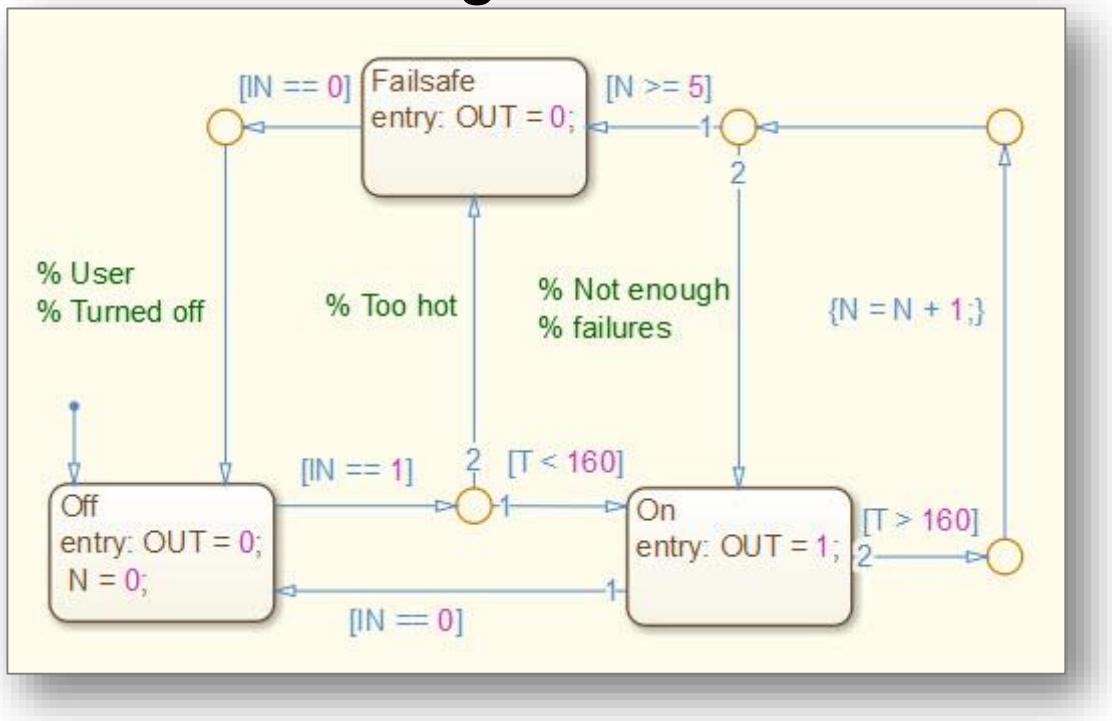
What is Stateflow?

- Stateflow is a state machine design environment

Draft



Design and Test

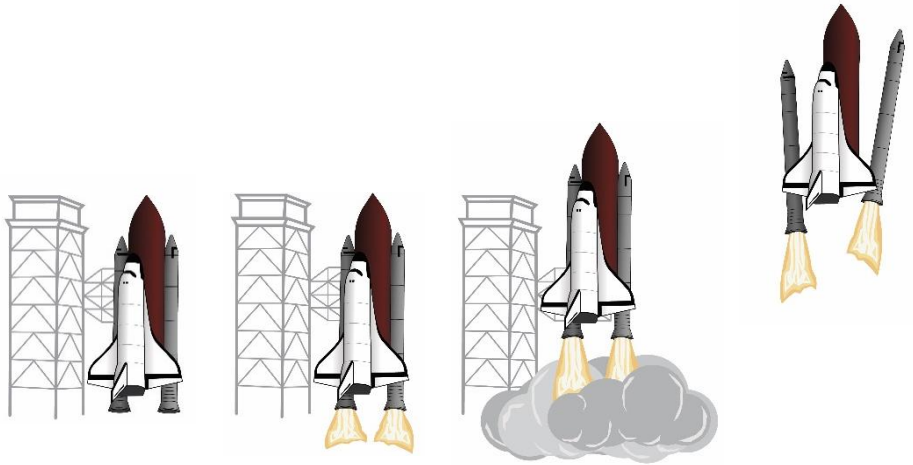


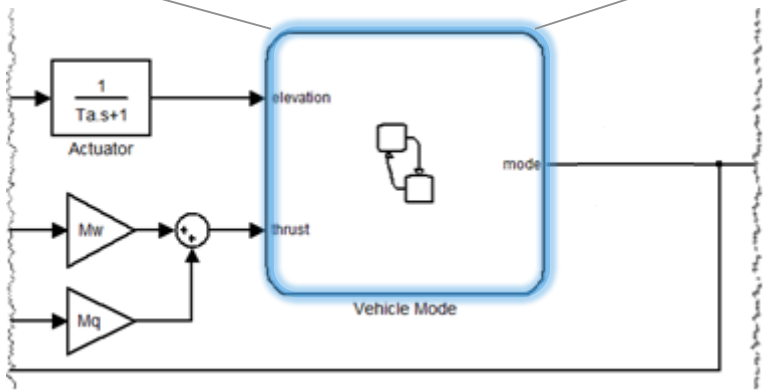
Why Stateflow?

Extend Simulink with state charts and flow graphs

Design supervisory control, scheduling, and mode logic

Model state discontinuities and instantaneous events





How Does Stateflow Work with Simulink?

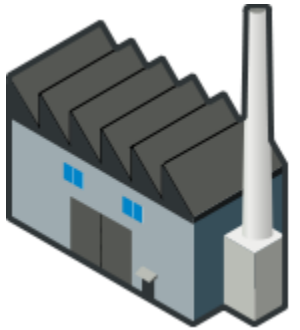
Simulink models **continuous** changes in dynamic systems.

Stateflow models **instantaneous** changes in dynamic systems.

Real-world systems have to respond to both continuous and instantaneous changes.

suspension dynamics
gear changes

propulsion system
liftoff stages

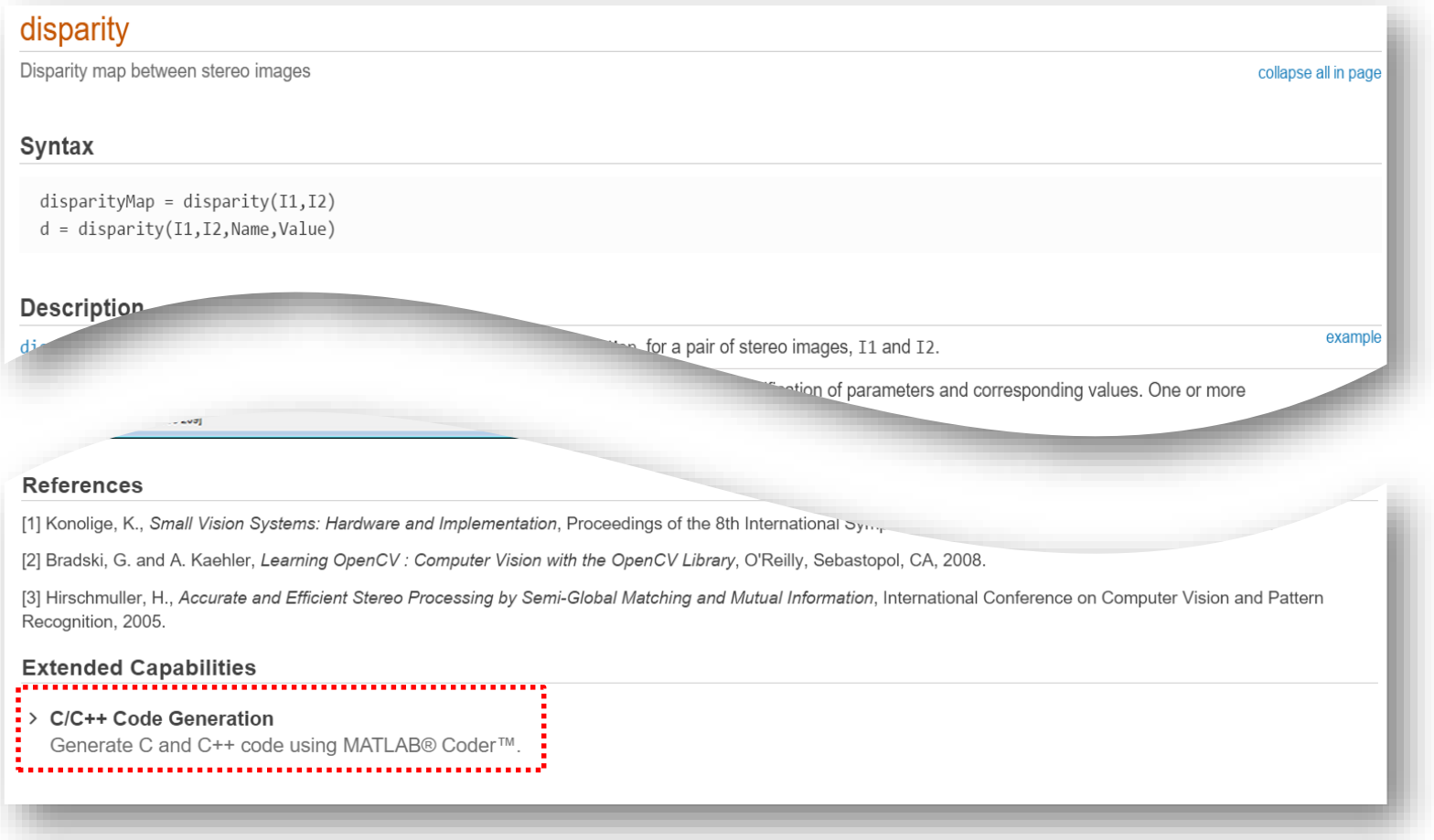


manufacturing robot
operation modes

Use both Simulink and Stateflow so that you can use the right tool for the right job.

Model-Based Design with MATLAB

- Limitations for code generation



disparity

Disparity map between stereo images

collapse all in page

Syntax

```
disparityMap = disparity(I1,I2)
d = disparity(I1,I2,Name,Value)
```

Description

example

for a pair of stereo images, I1 and I2.

ation of parameters and corresponding values. One or more

References

[1] Konolige, K., *Small Vision Systems: Hardware and Implementation*, Proceedings of the 8th International Sy...

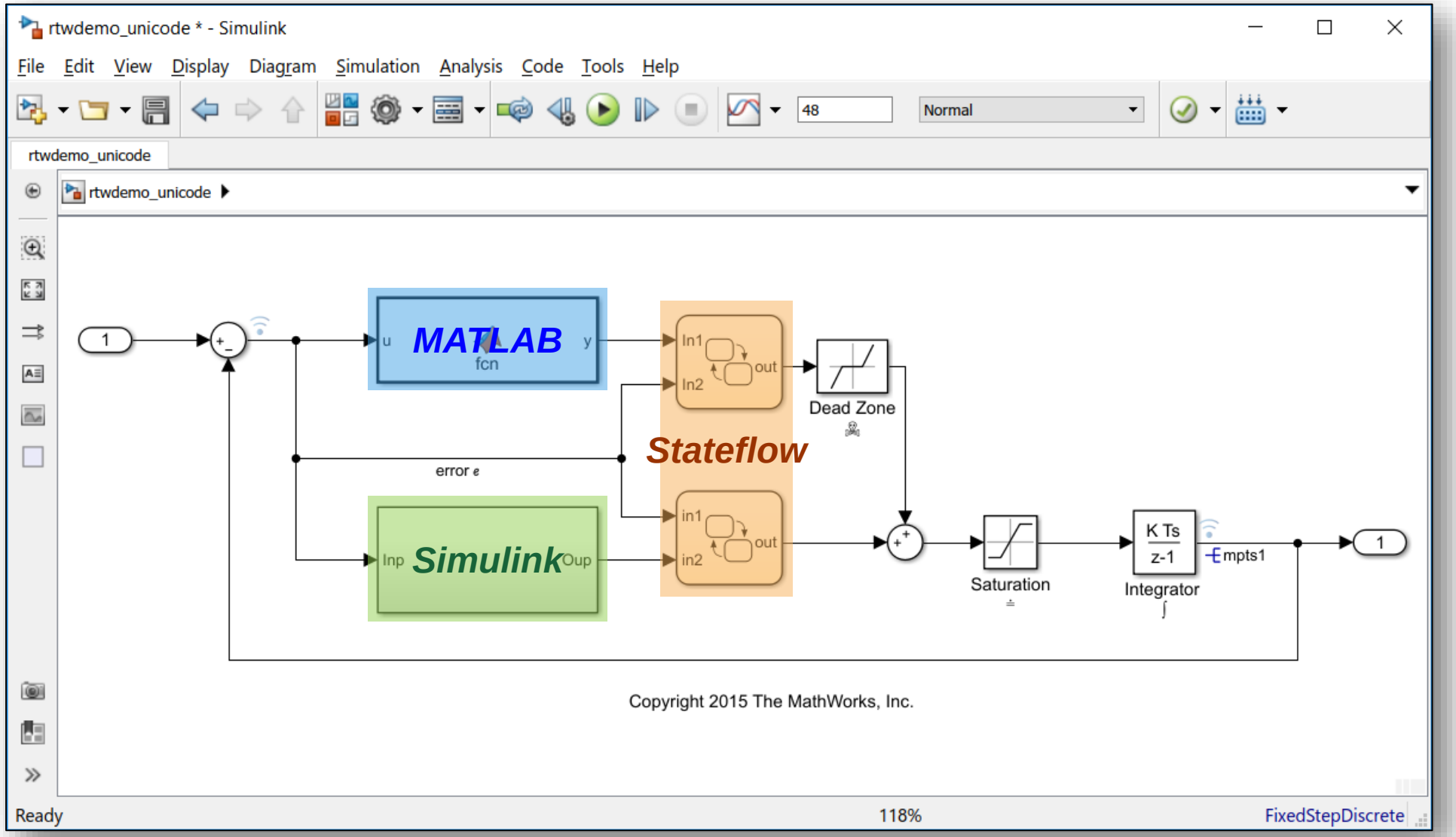
[2] Bradski, G. and A. Kaehler, *Learning OpenCV : Computer Vision with the OpenCV Library*, O'Reilly, Sebastopol, CA, 2008.

[3] Hirschmuller, H., *Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information*, International Conference on Computer Vision and Pattern Recognition, 2005.

Extended Capabilities

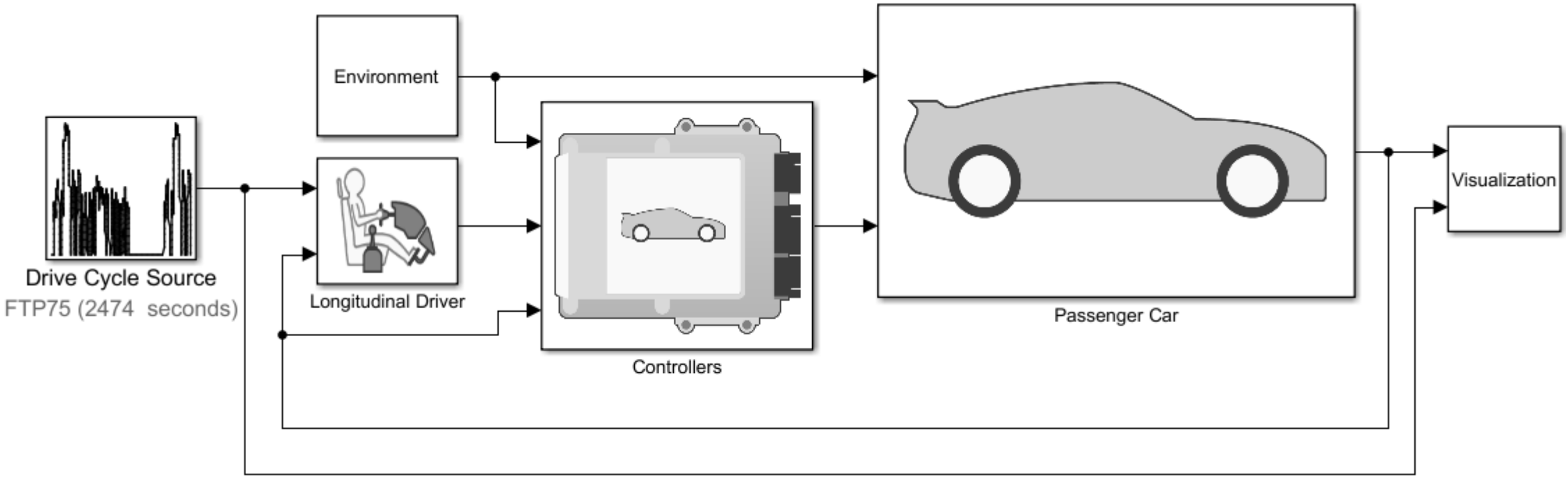
> **C/C++ Code Generation**
> Generate C and C++ code using MATLAB® Coder™.

Software Modeling Example



Model-Base Design for Vehicle Applications

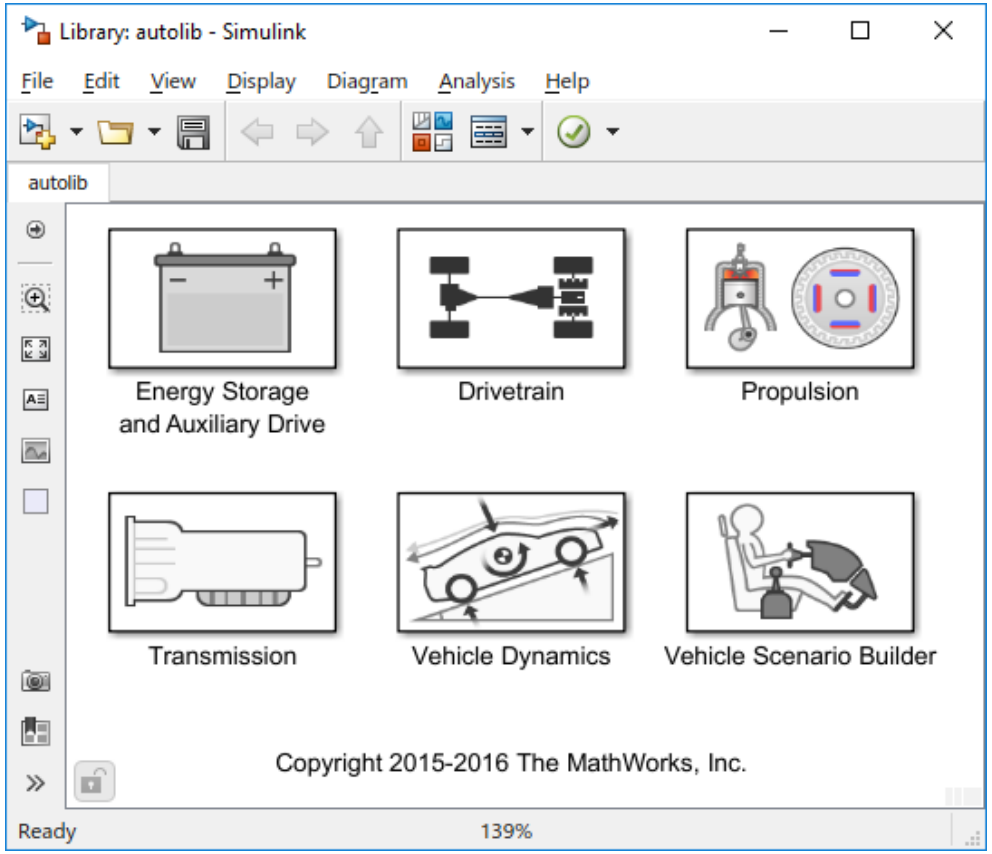
MathWorks Solution for Full Vehicle Simulation



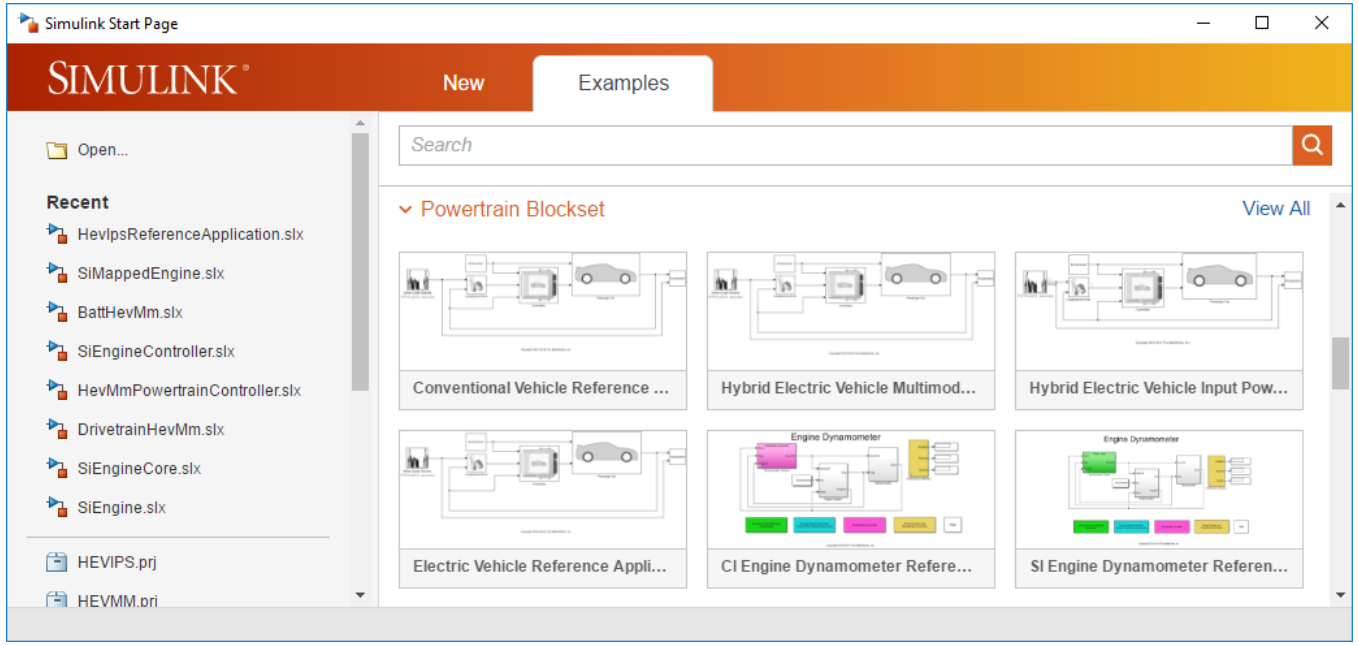
Lower the barrier to entry for Model-Based Design

Powertrain Blockset Features

Library of blocks



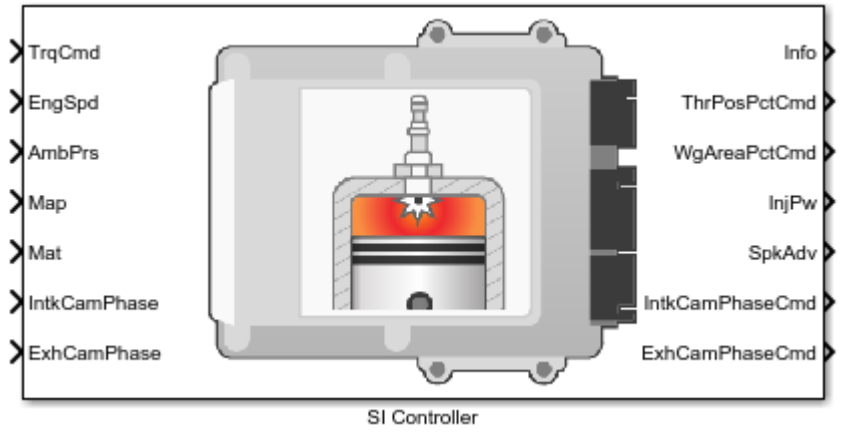
Pre-built reference applications



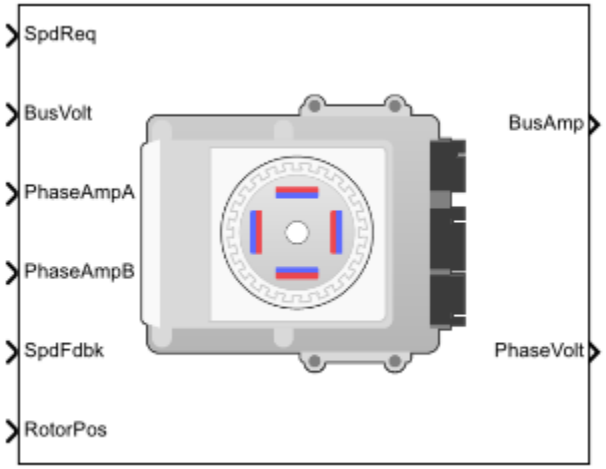
- Conventional (Spark Ignition / Compression Ignition) vehicle
- Electric vehicle (EV)
- Multi-mode / Power Split / P2 hybrid electric vehicle (HEV)
- Virtual engine dynamometer system

Library: Controllers

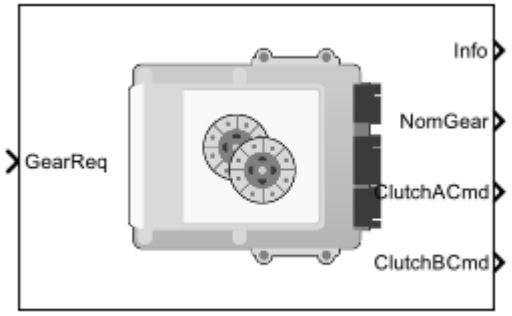
- Libraries include basic component and subsystem controllers
  - Powertrain Control Module (PCM)
  - Hybrid Control Module (HCM)
  - Engine Control Module (ECM)
  - Transmission Control Module (TCM)
  - Component controllers
  - State estimators
- Like plant models, the controllers are open and reconfigurable
- Provides realistic starting point for your own controller development



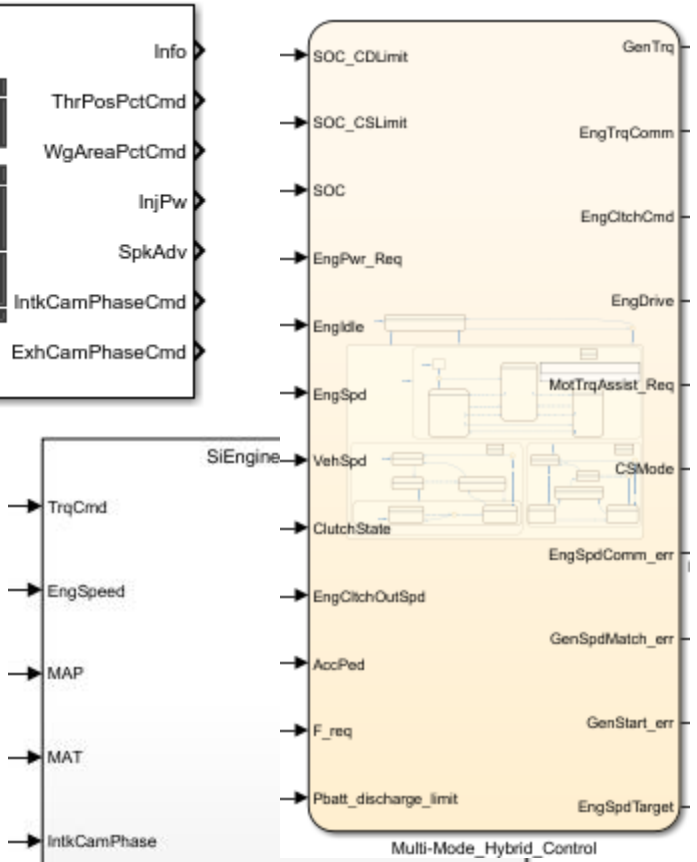
SI Controller



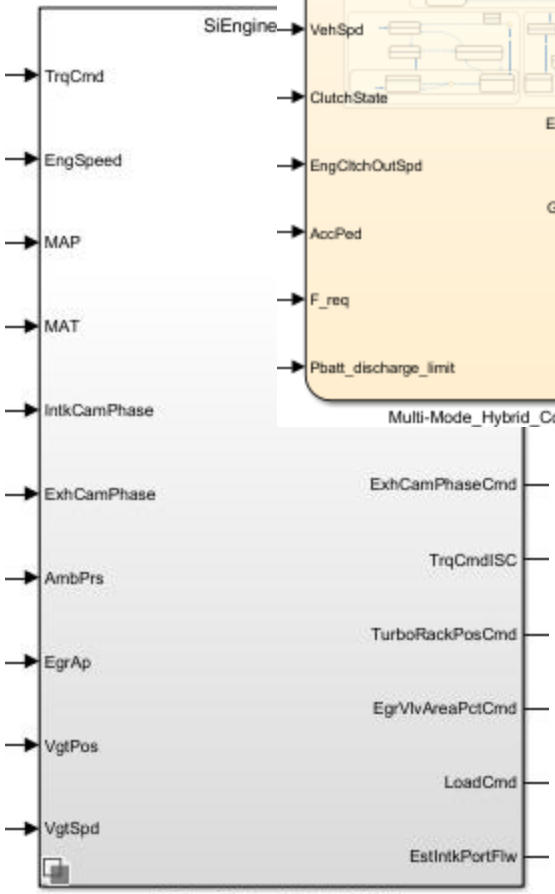
Interior PM Controller



DCT Controller

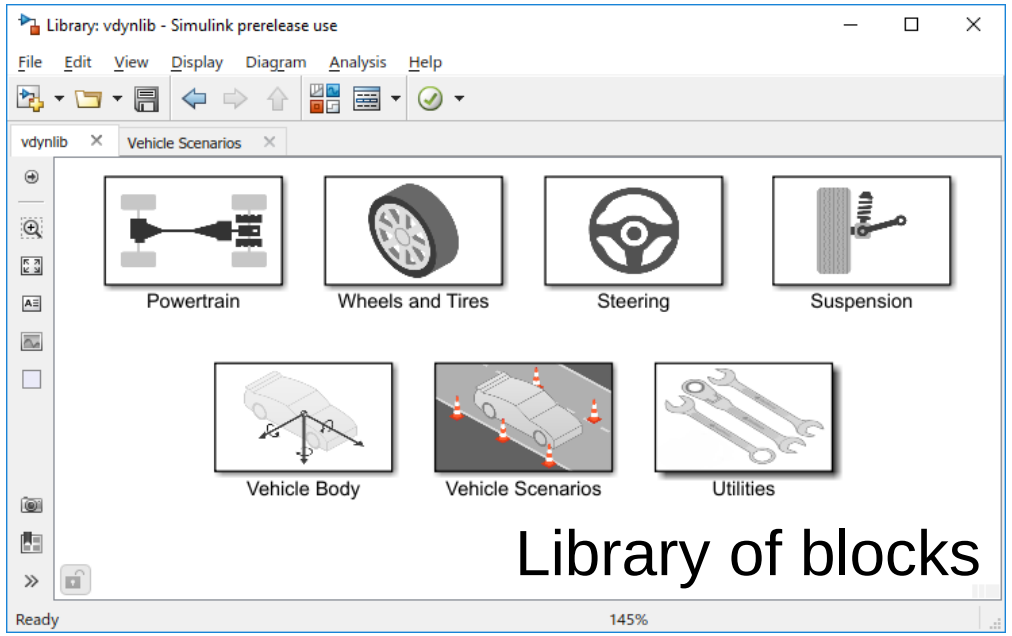


Multi-Mode_Hybrid_Control



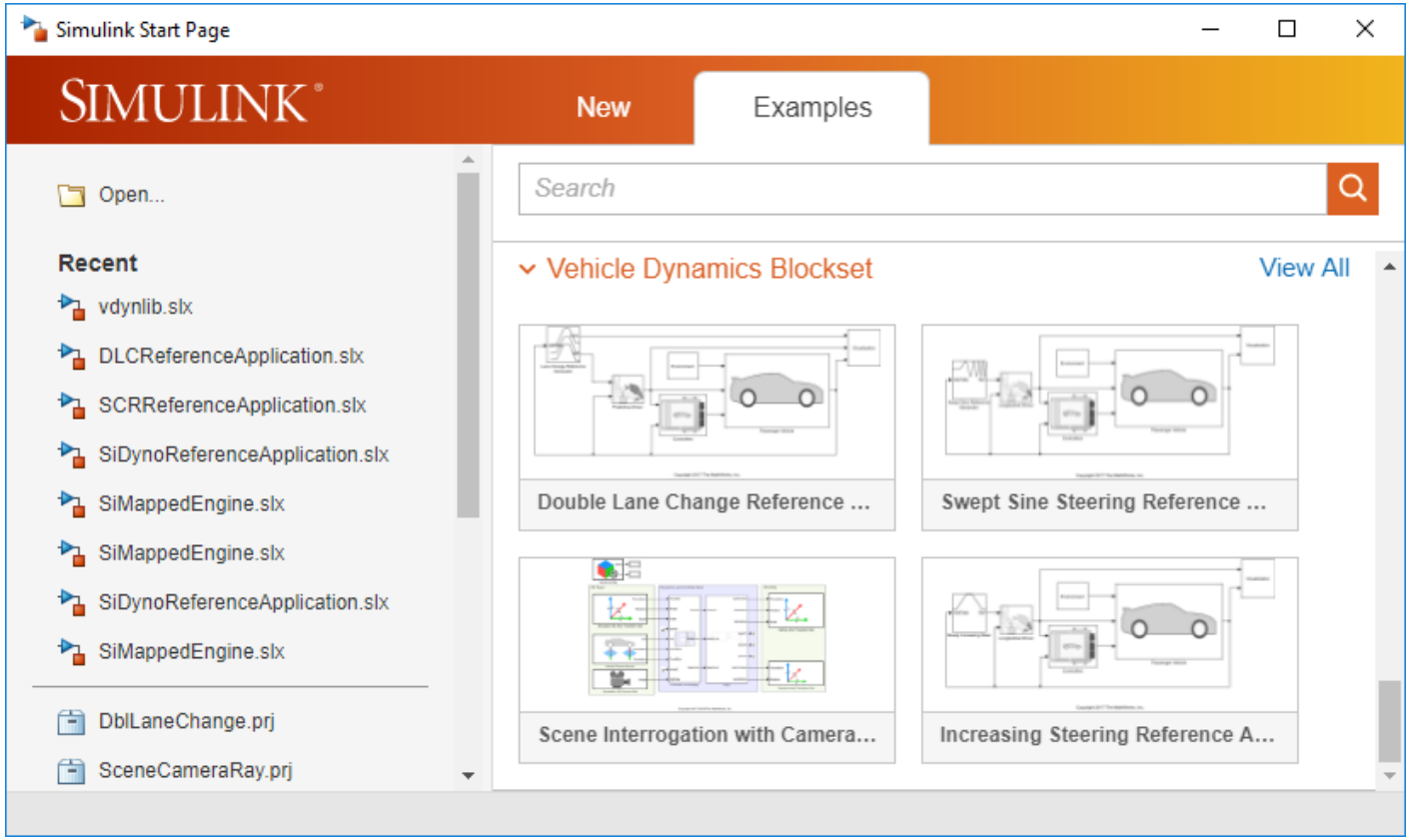
Engine Control Module (ECM)

Vehicle Dynamics Blockset Features



Library of blocks

Pre-built reference applications



Vehicle Dynamics Blockset

R2018a

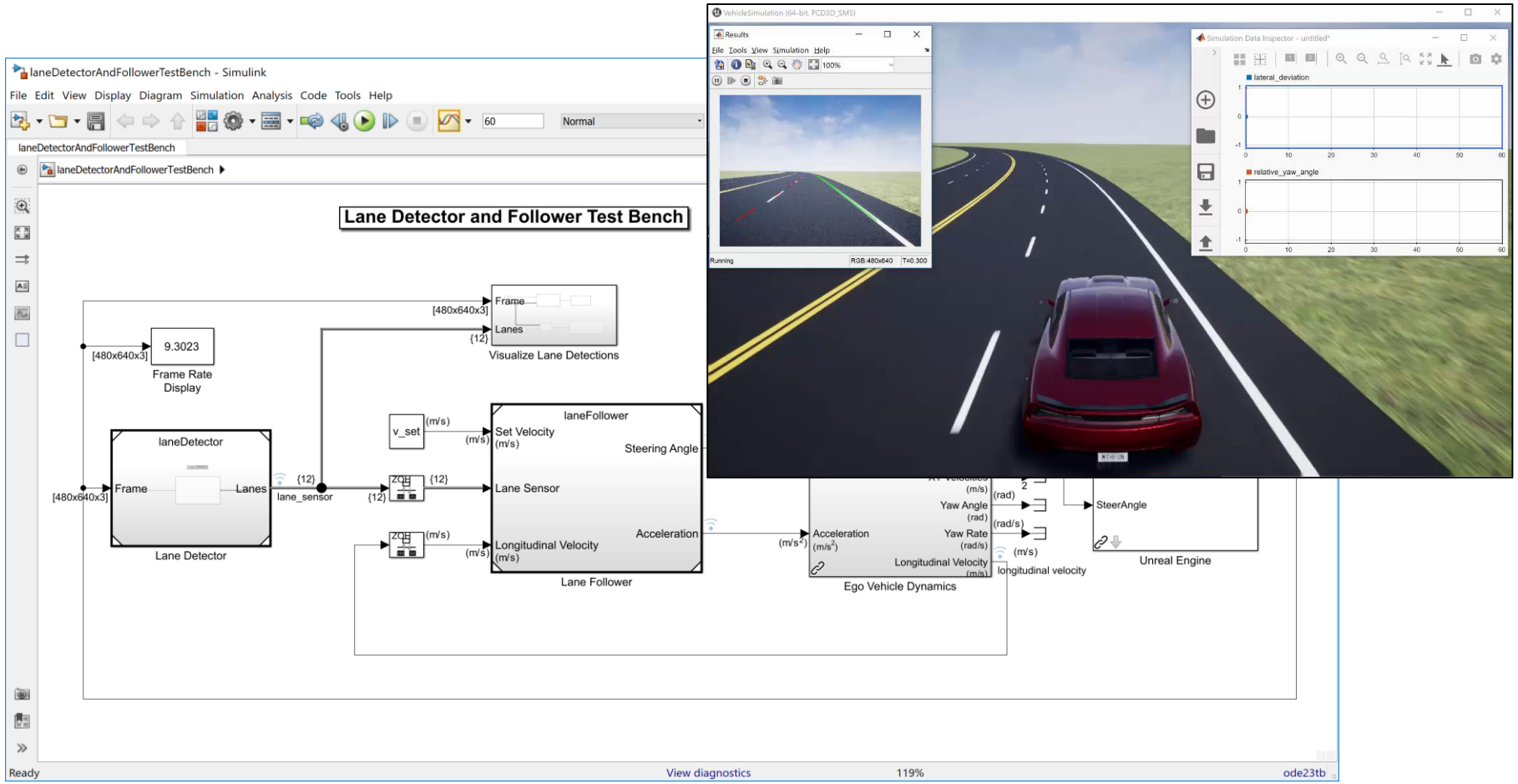
- Model and simulate vehicle dynamics in a virtual 3D environment
- Use Vehicle Dynamics Blockset for:
  - Ride & handling: characterize vehicle performance under standard driving maneuvers
  - Chassis controls: design and test chassis control systems
  - ADAS / AD: create virtual 3D test ground for ADAS and automated driving features

Ride & handling

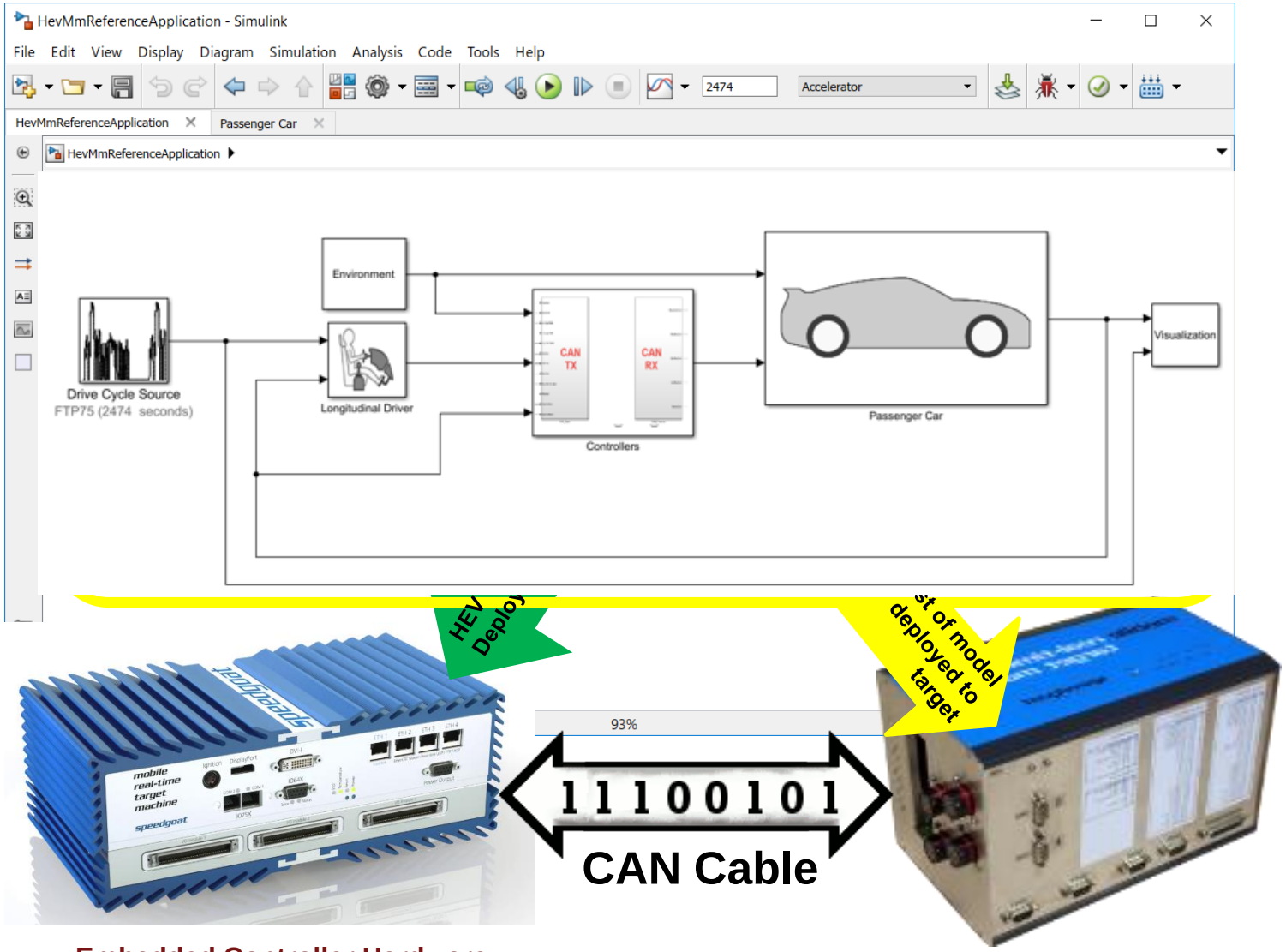
Chassis controls

ADAS / AD

Lane detector and follower system test bench simulates vehicle dynamics with Unreal Engine to synthesize camera images



HIL Testing



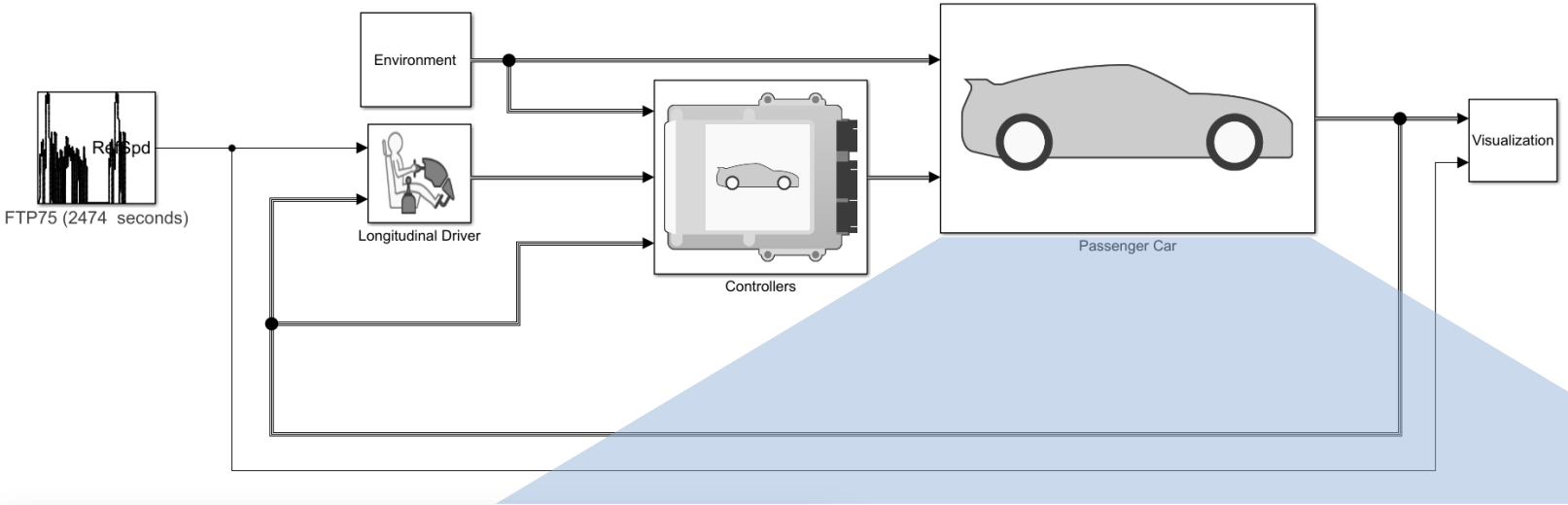
Speedgoat Rapid Control Prototyping System

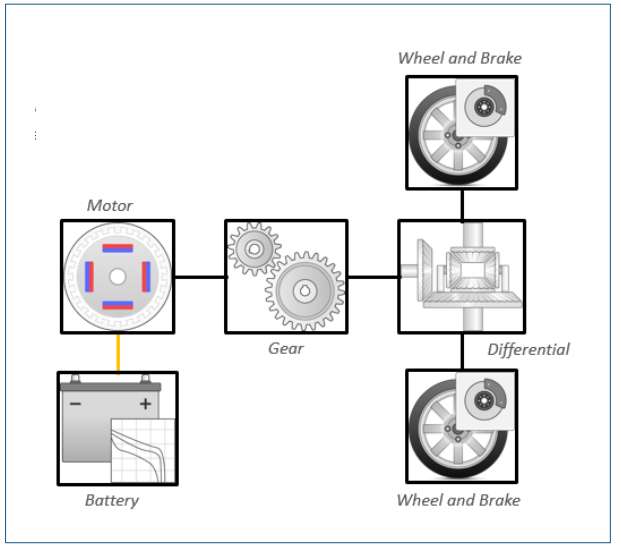
Embedded Controller Hardware

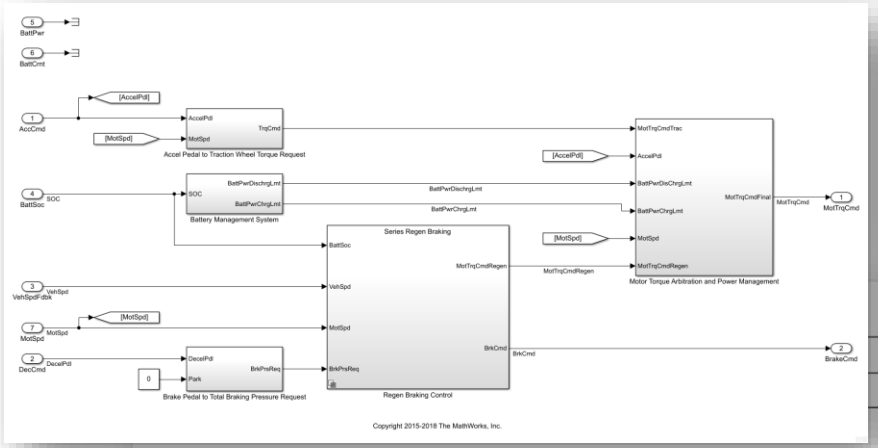
Speedgoat Hardware in-the-loop System

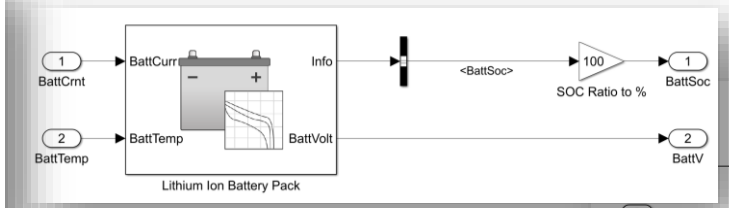
Target Computer Hardware

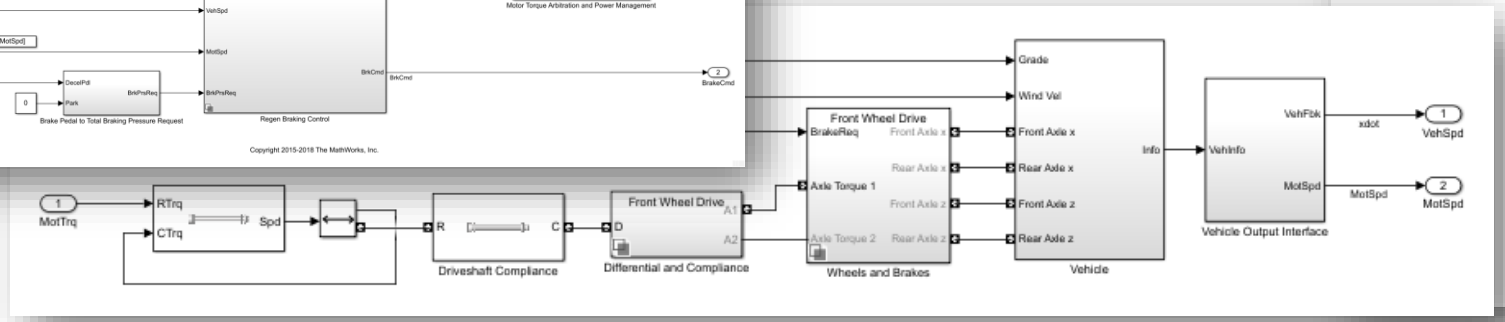
EV Modeling Example

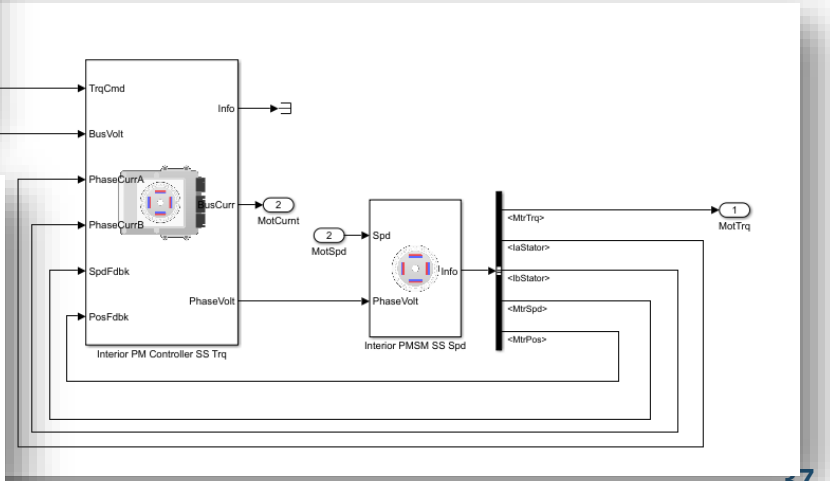




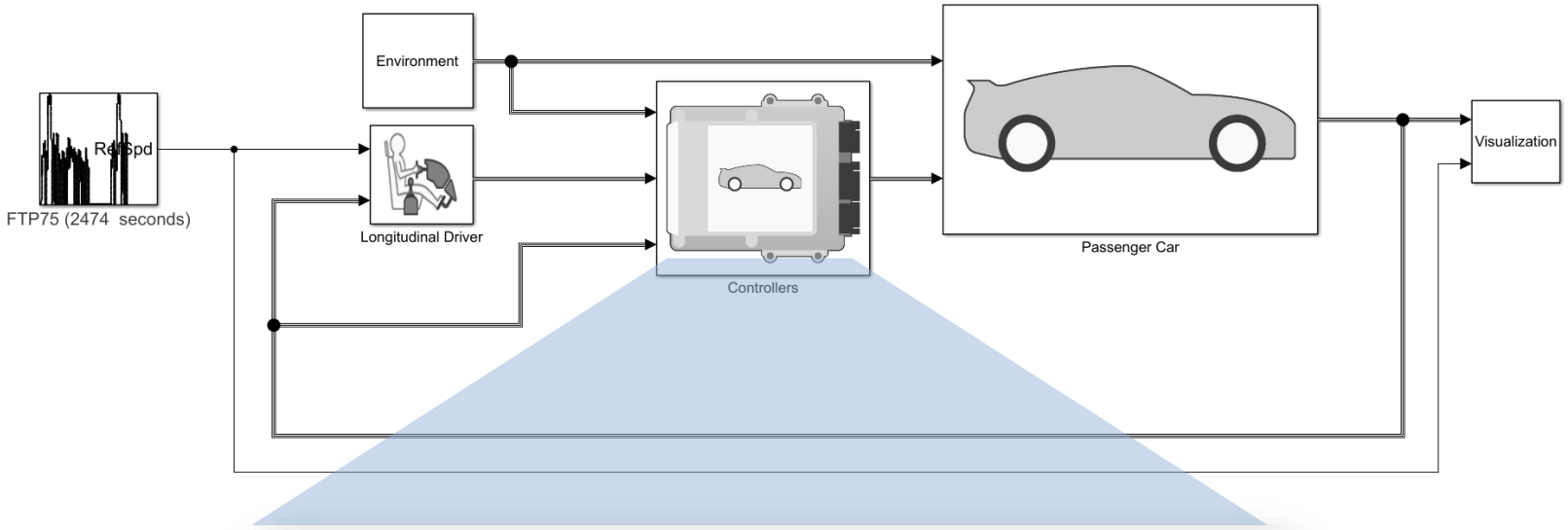


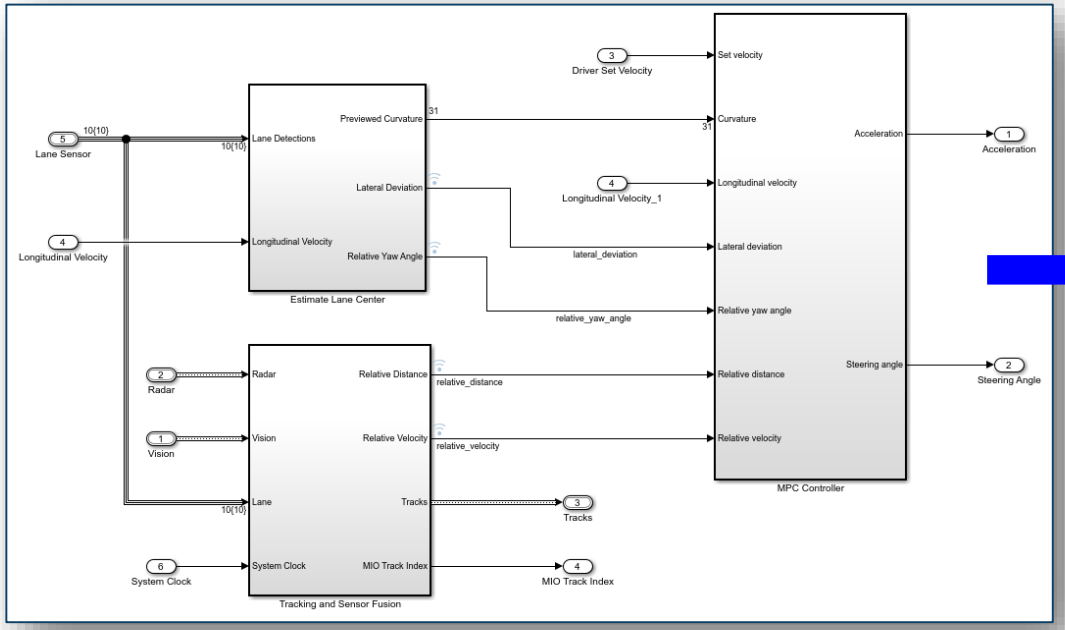




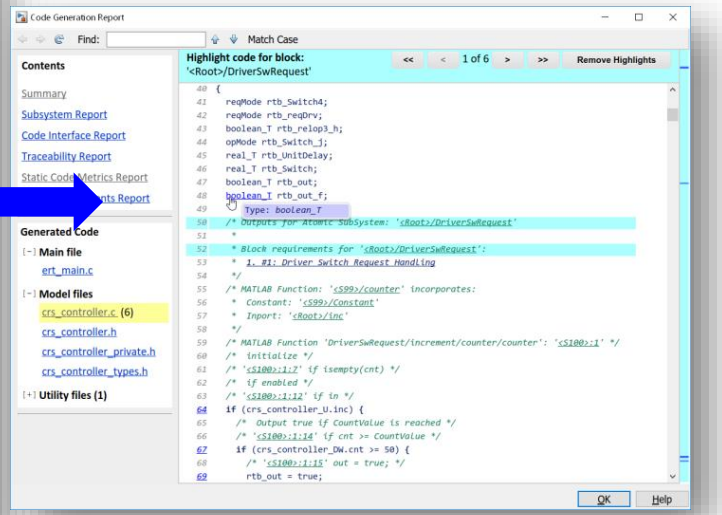


Controller Design

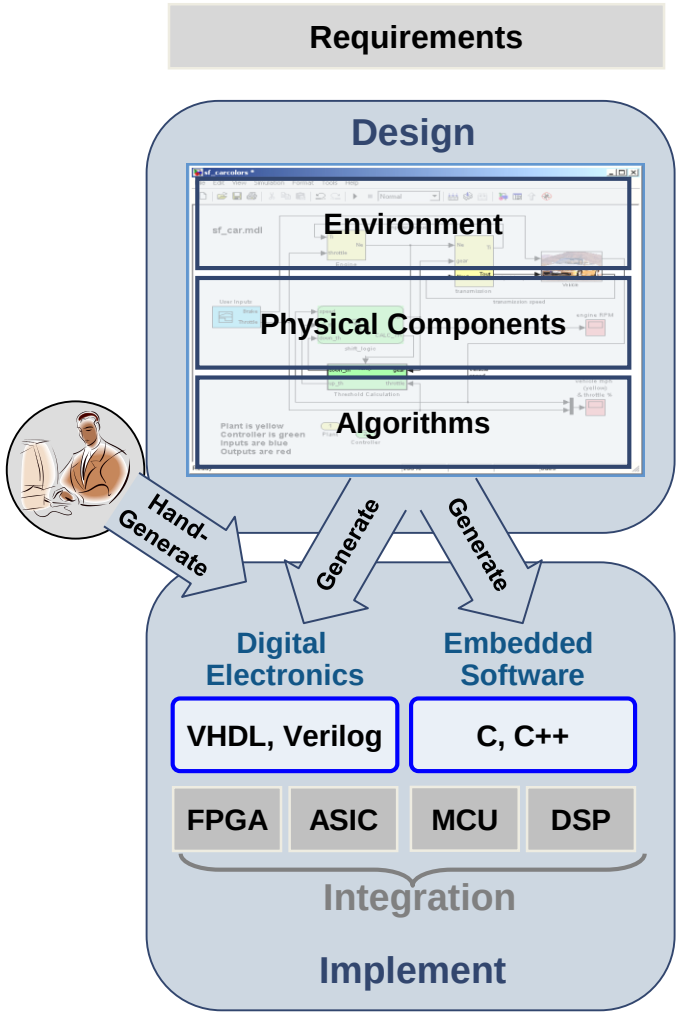




Automatic code generation



Code Generation Overview



- ISO ANSI-C production code generation with Embedded Coder
- Use of Legacy Code Tool to make a call to existing C code in generated code from Simulink models
- HTML Report Generation for Review and Traceability

Code Generation Products

- **MATLAB Coder**

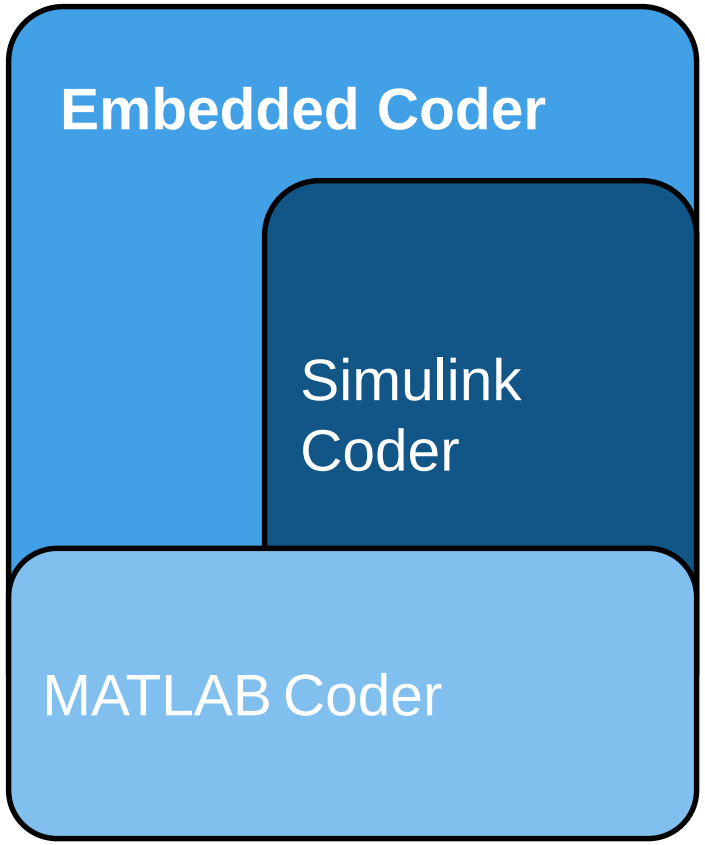
  - Generate C and C++ code from MATLAB code

- **Simulink Coder**

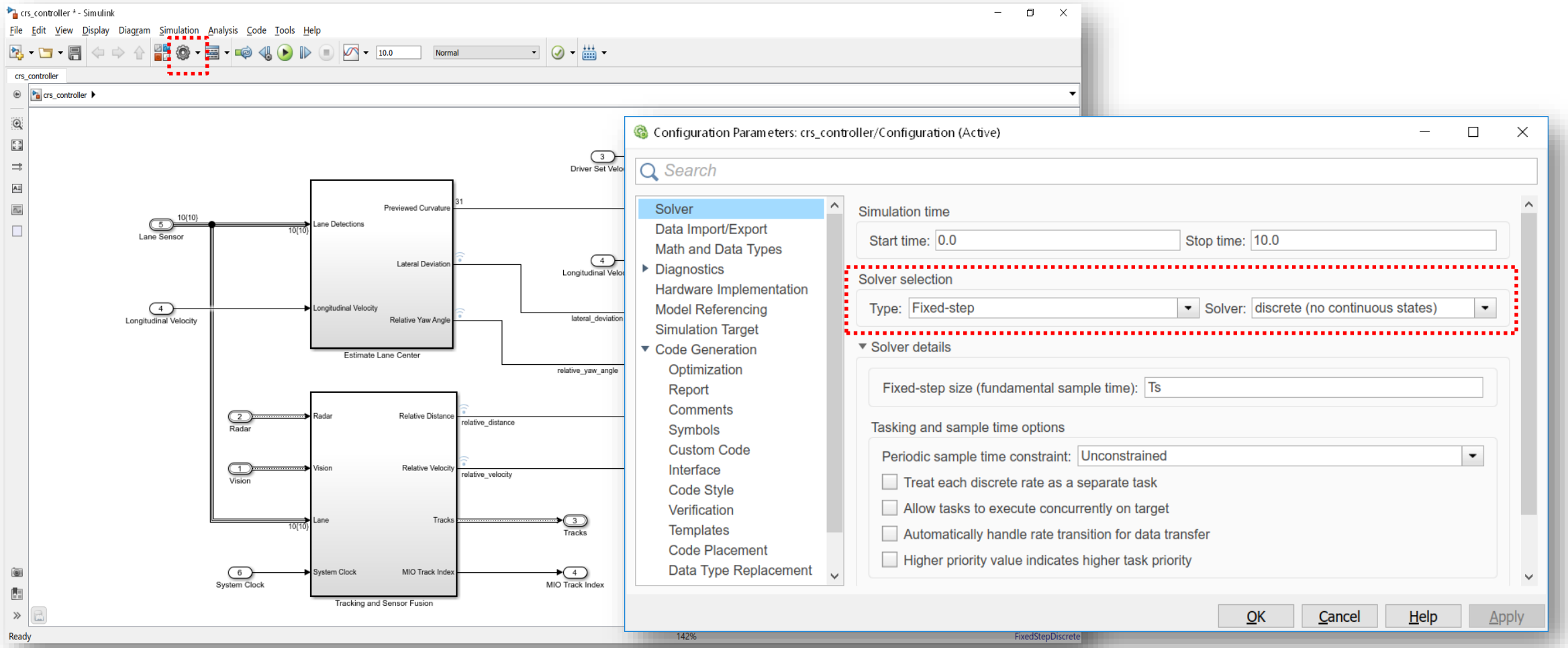
  - Generate C and C++ code from Simulink and Stateflow models

- **Embedded Coder**

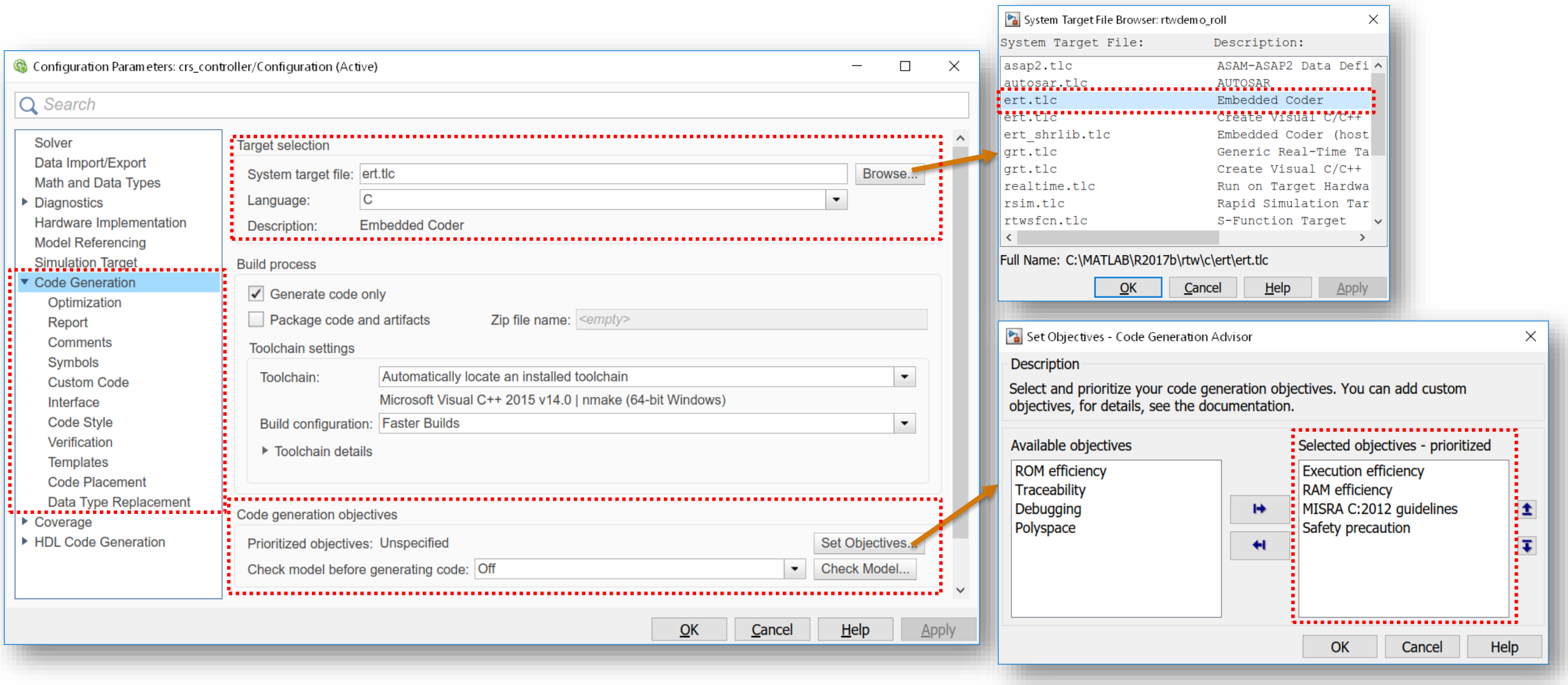
  - Generate C and C++ code optimized for embedded systems



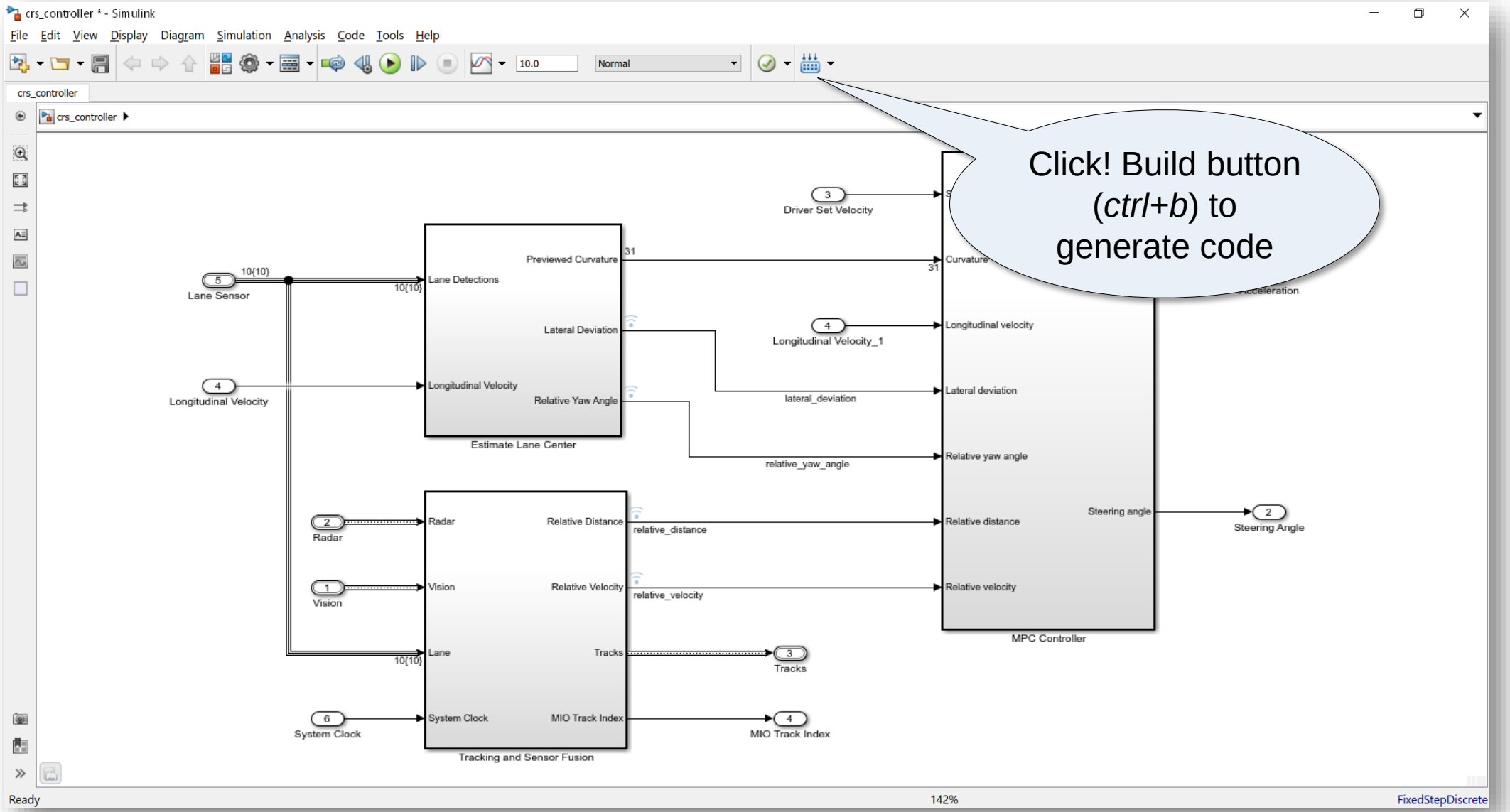
Prepare a Model for Embedded Code Generation



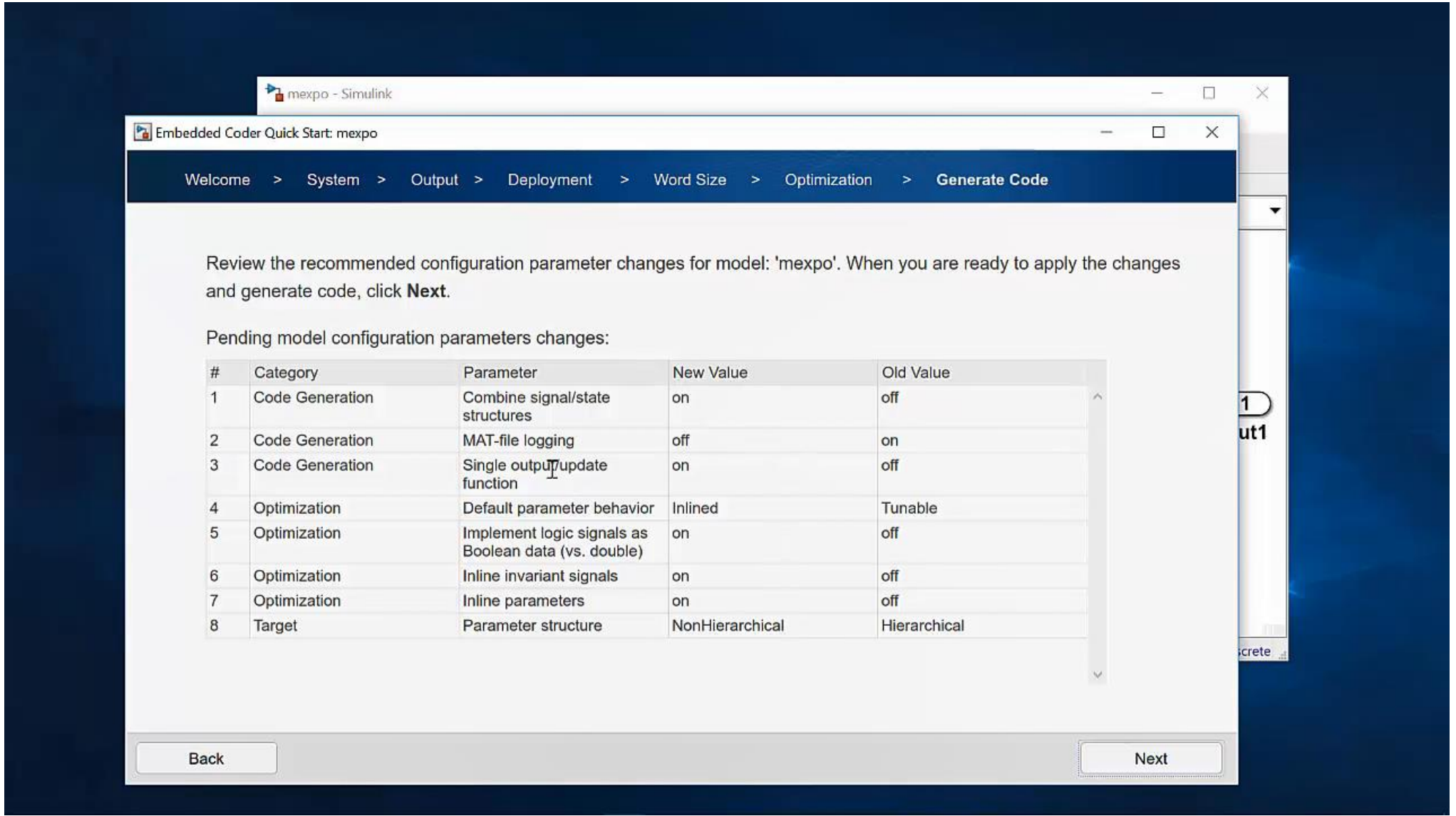
Prepare a Model for Embedded Code Generation



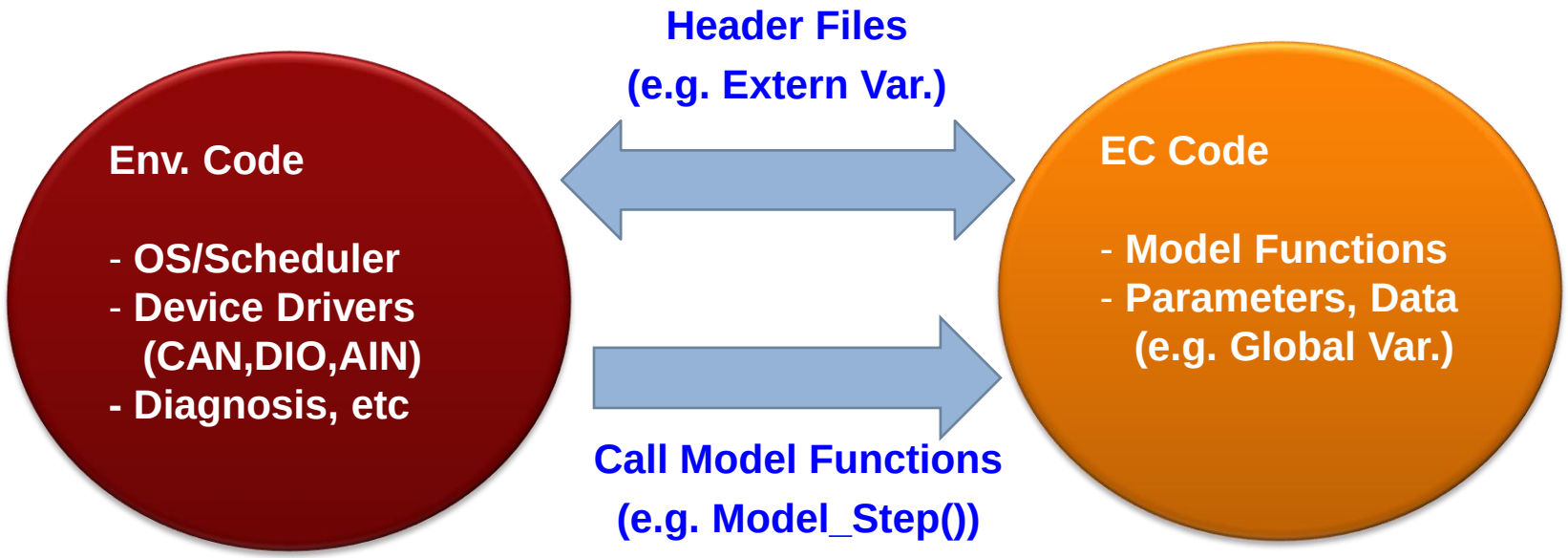
Generating Code



Demo: Embedded Coder Quick Start



Interfacing Generated Code



- Share entry points, data and parameters with header files
- Call generated model functions from OS/Scheduler

Implementation

Example : Integrating Generated Controller Code with an Embedded Software Project

Embedded Software Project

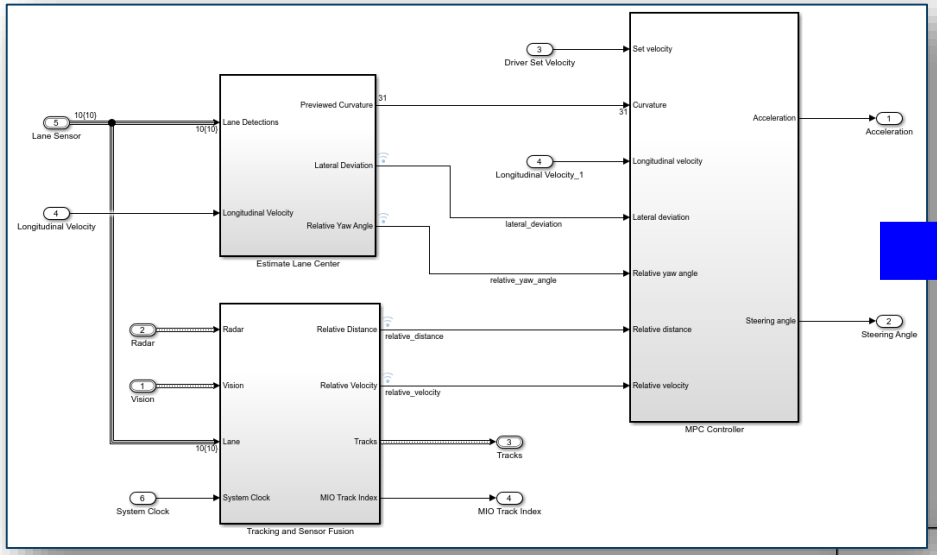
Main()

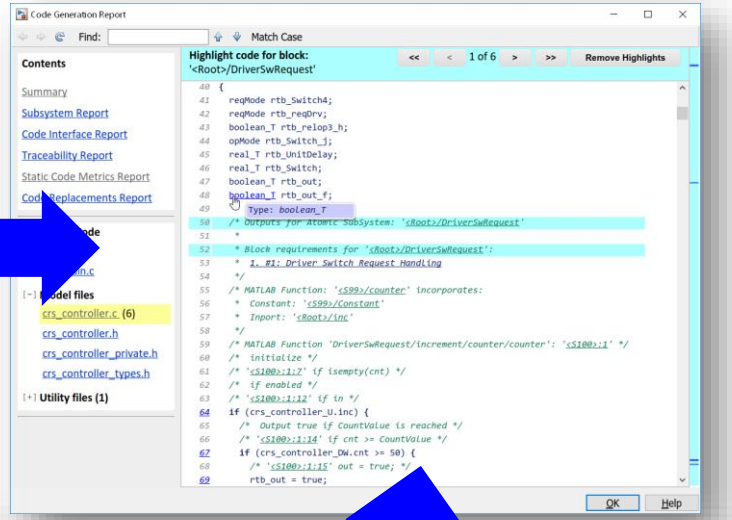
```
{
    Commands_Init
    PWM_Init
    ADC_Init
    Encoder_Init

    Controller_Init

    while(1) { }
}
```

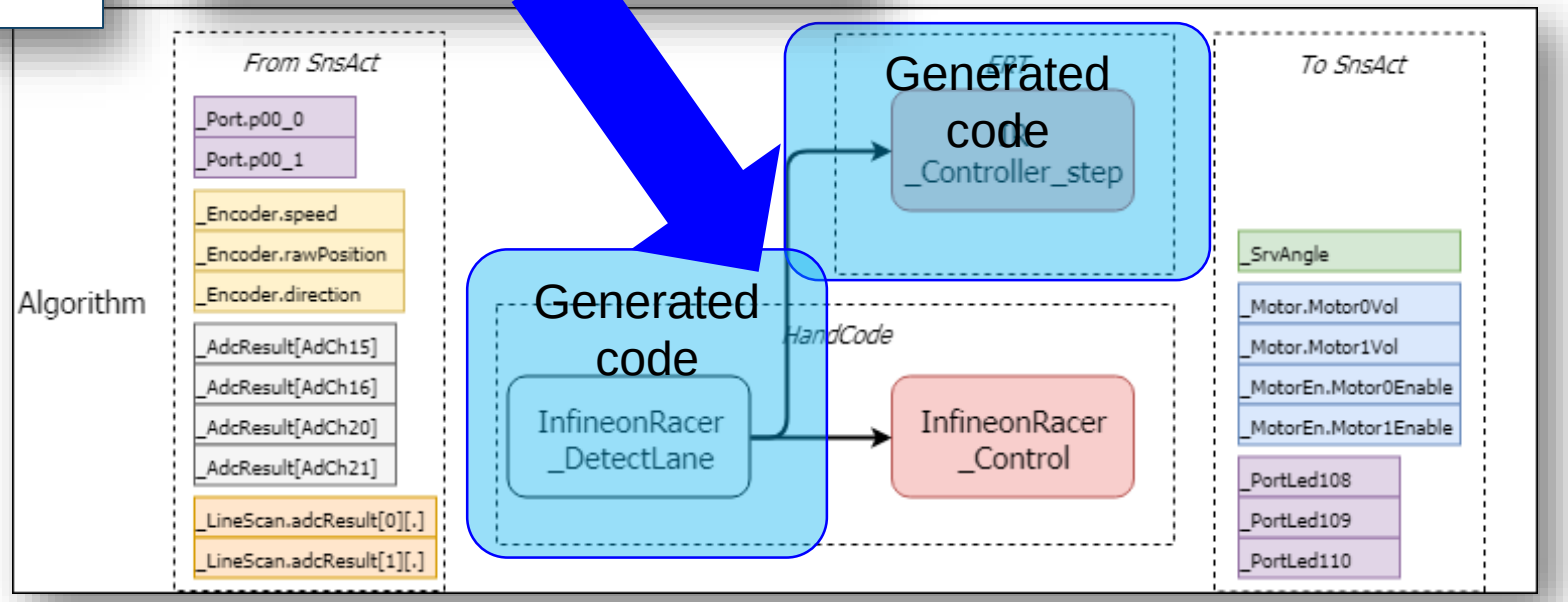
ADC_InterruptServiceRountine_25kHz()

```
{
    Commands_Read
    ADC_Read
    Encoder_Read

    Controller_Step

    PWM_Write
}
```

Software Integration Example for SMCC

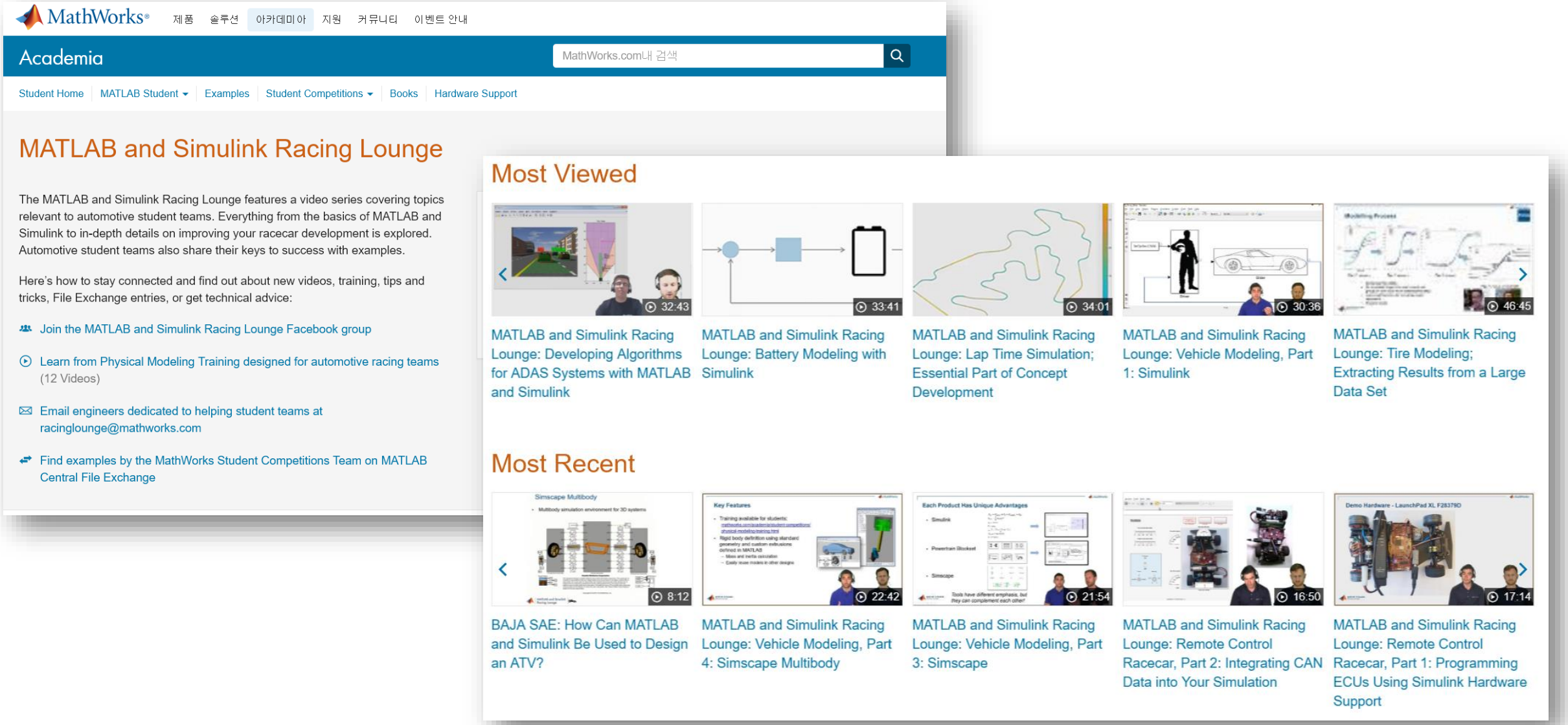






https://aurixracer.readthedocs.io/ko/latest/SimulinkAsAProgrammingLanguage/index.html

Self-study Resources



https://kr.mathworks.com/academia/student-competitions/racing-lounge.html?s_tid=srchtitle

Self-Study Resources for Embedded Code Generation

- Embedded Coder Quick Start Video
  https://kr.mathworks.com/videos/coder-summit-2018-how-to-generate-production-code-in-5-minutes--1522057622892.html?s_tid=srchtitle
- Simulink와 Embedded Coder를 이용한 최적 코드 생성
  http://www.matlabexpo.com/kr/2017/proceedings/better-than-hand-generating-highly-optimized-code-using-simulink-embedded-coder.pdf
- C code generation from Simulink model(webinar)
  https://kr.mathworks.com/videos/software-design-and-c-code-generation-using-simulink-116860.html?elqsid=1524127259550&potential_use=Commercial
- Other Embedded Coder Videos
  https://kr.mathworks.com/products/embedded-coder/videos.html?s_tid=srchtitle

MathWorks

지원

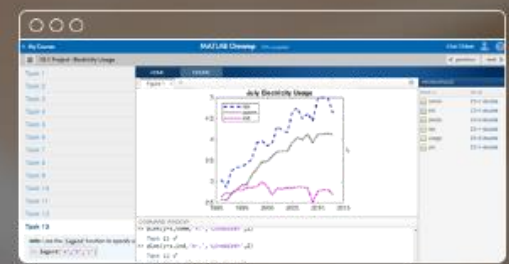
Search Support 지원

MATLAB 및 Simulink 튜토리얼 학습하기

온라인 자기 주도형 교육 과정에서 MATLAB 을 직접 사용해보실 수 있습니다.

- 웹 브라우저에서 프로그래밍 연습
- 짧은 비디오 데모
- 즉각적인 피드백

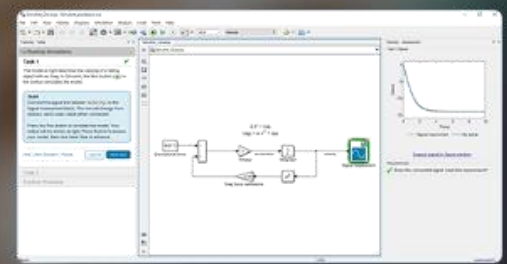
모든 과정 둘러보기



MATLAB Onramp

무료 - MATLAB의 필수 기능을 학습합니다.

시작



Simulink Onramp

무료 - Simulink의 필수 기능을 빠르게 학습합니다.

제품정보

MATLAB 기본 배우기

MATLAB® 은 전 세계 수백만 명의 엔지니어와 과학자가 이용하는 고급 언어(high-level language)이자 대화형 환경입니다. 신호 및 이미지 처리, 통신, 제어 시스템 및 계산 금융을 포함한 여러 분야에서 아이디어를 시각화할 수 있도록 돕습니다.

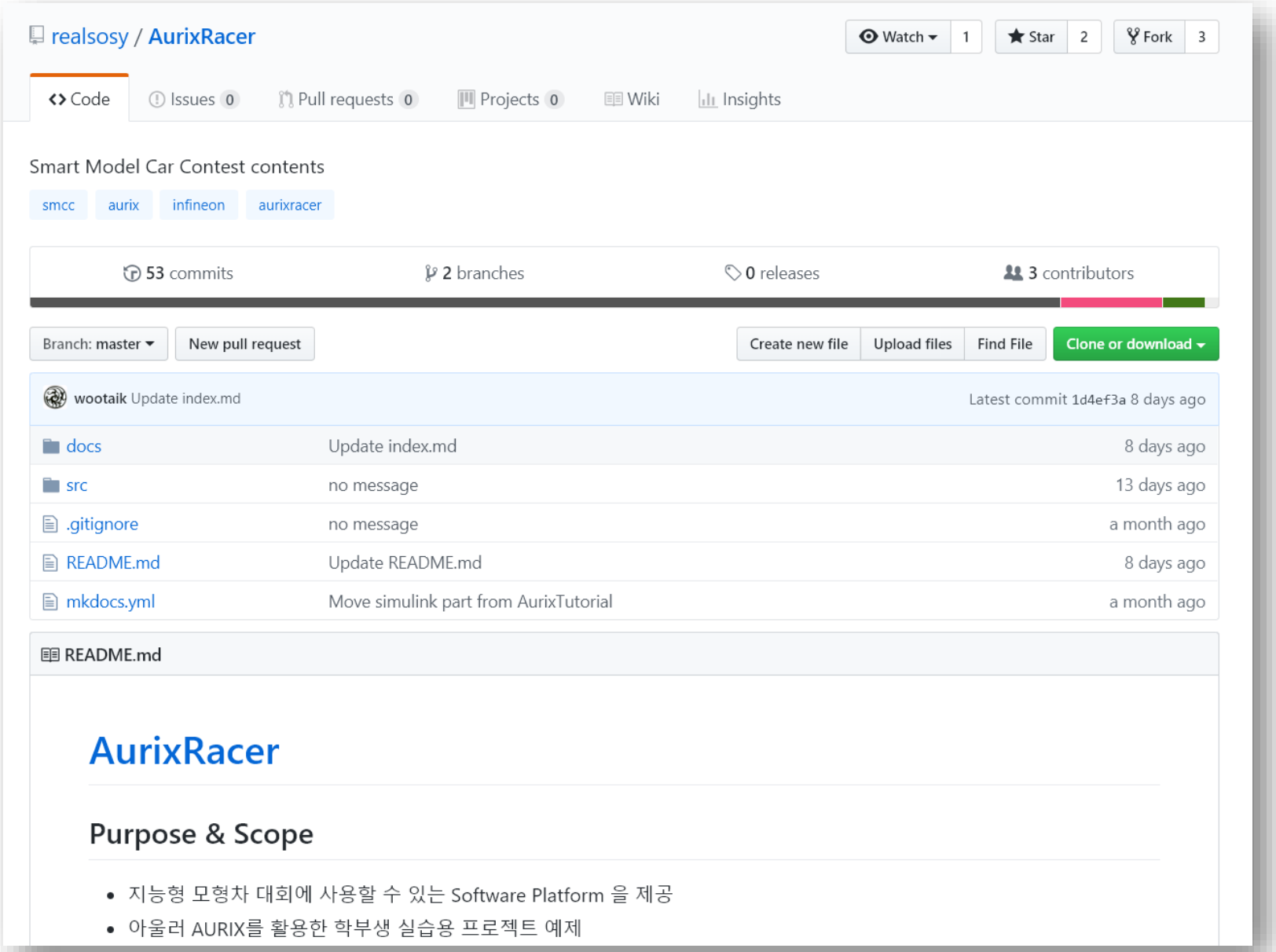
Simulink 기본 배우기

Simulink® 는 블록 다이어그램 환경으로 구성된 멀티도메인 시뮬레이션 및 모델 기반 설계를 위한 도구입니다. 시뮬레이션, 자동 코드 생성과 임베디드 시스템의 지속적인 테스트 및 검증을 지원합니다. 그래픽 편집기, 사용자 정의 가능한 블록 라이브러리, 모델링을 위한 솔버를 활용하는 동적 시스템을 시뮬레이션합니다.

https://kr.mathworks.com/support/learn-with-matlab-tutorials.html

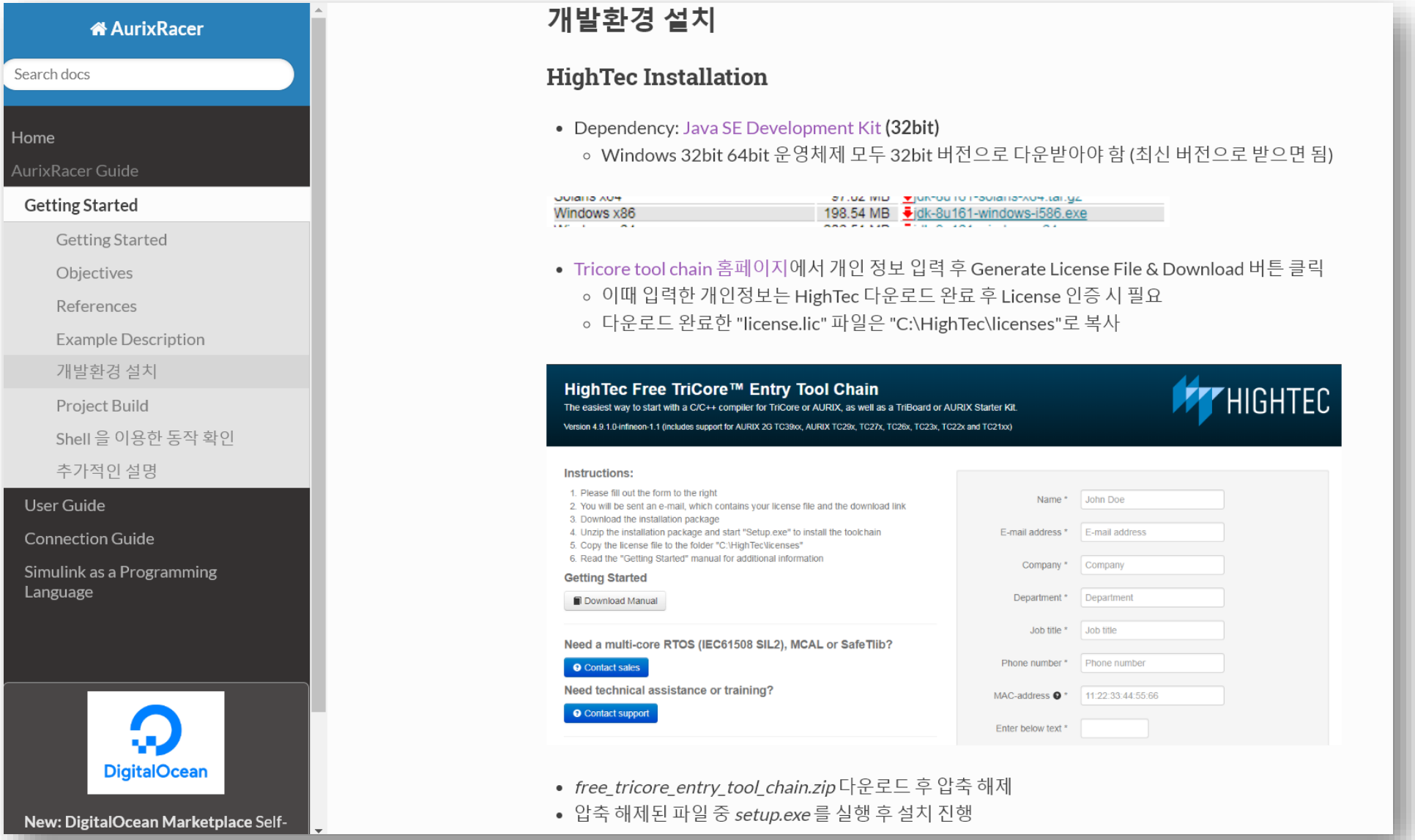
Prerequisites for Development Environment

AurixRacer 다운로드



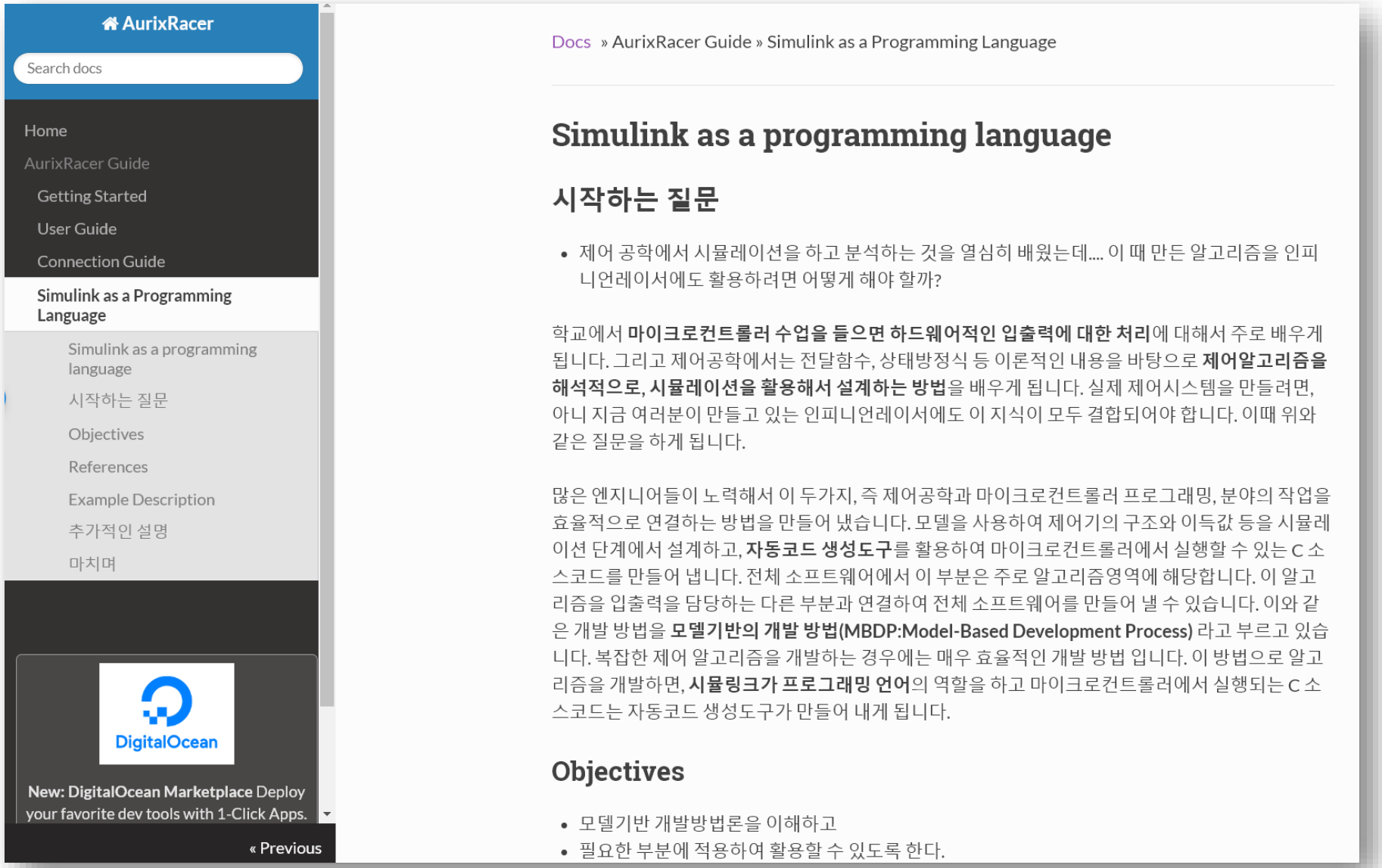
https://github.com/realsosy/AurixRacer

개발환경 설치



https://aurixracer.readthedocs.io/ko/latest/GettingStarted/index.html#_1

시뮬링크 개발 환경 및 자동 생성 코드 사용 방법 가이드



https://aurixracer.readthedocs.io/ko/latest/SimulinkAsAProgrammingLanguage/index.html#_5

New Development Platform (Optional)

Hitex AURIX Target for Simulink

Simulink Blockset for targeting control algorithms onto Infineon AURIX MCUs

Highlights

- Enables Simulink® users to easily target Infineon AURIX™ MCUs with their models
- Supports MathWorks® code generation via Embedded Coder®
- Allows users to quickly move existing algorithms into AURIX™ applications
- Full support for the Hitex ShieldBuddy TC275
- Multicore operation supports full power of the AURIX TC275 MCU

Description

Infineon's AURIX™ family of microcontrollers serves the precise needs of the automotive industry in terms of performance and safety. The innovative multicore architecture, based on up to three independent 32-bit TriCore™ CPUs, has been designed to meet the highest safety standards, while increasing performance significantly. Using the AURIX platform, automotive developers will be able to control powertrain (including hybrid and electrical vehicles) and safety applications (such as steering, braking, airbag, and advanced driver assistance systems) with a single MCU platform. Developing on AURIX can reduce MCU safety development time by 30%, while maintaining power consumption of a single-core microcontroller and leveraging a performance surplus of 50-100% for more functionality now and in the future. AURIX also protects IP

Required Products

- MATLAB
- Simulink
- Embedded Coder
- MATLAB Coder
- Simulink Coder

Recommended Products

- Fixed-Point Designer
- Stateflow

Platforms

- Windows

Support

- Consulting
- E-mail
- System integration
- Telephone
- Training

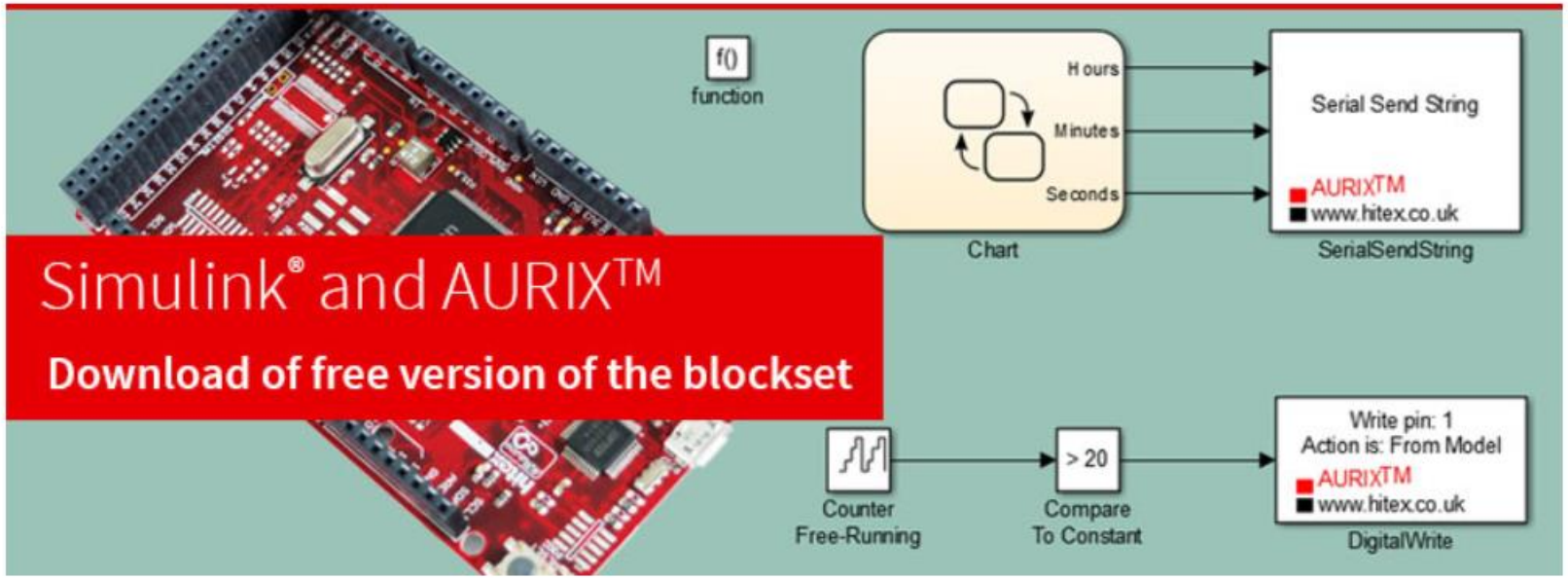
Product Type

- Embedded Hardware - MCU, DSP, FPGA
- Embedded Software - Tools, IDE,

https://kr.mathworks.com/products/connections/product_detail/hitex-aurix-target.html

hitex
EMBEDDED TOOLS & SOLUTIONS

Building a safe and secure embedded world



Simulink® and AURIX™: Download free version of blockset!

Simulink is well established as a modelling tool particularly for Automotive and Aerospace applications; Hitex (UK) Ltd. has now developed a free blockset that supports the ShieldBuddy TC275. This blockset will allow engineers to easily reuse and port existing control algorithm models to AURIXTM without the need for major source code rework. Multicore operation is also supported in the paid version to leverage the full power of the AURIX TC275. Learn more and download the Free blockset.

You will get an email with the download link after submission.

Email *

seongyon@gmail.com

1) 소프트웨어 및 라이센스 받을 이메일 계정 입력

Submit

2) 제출

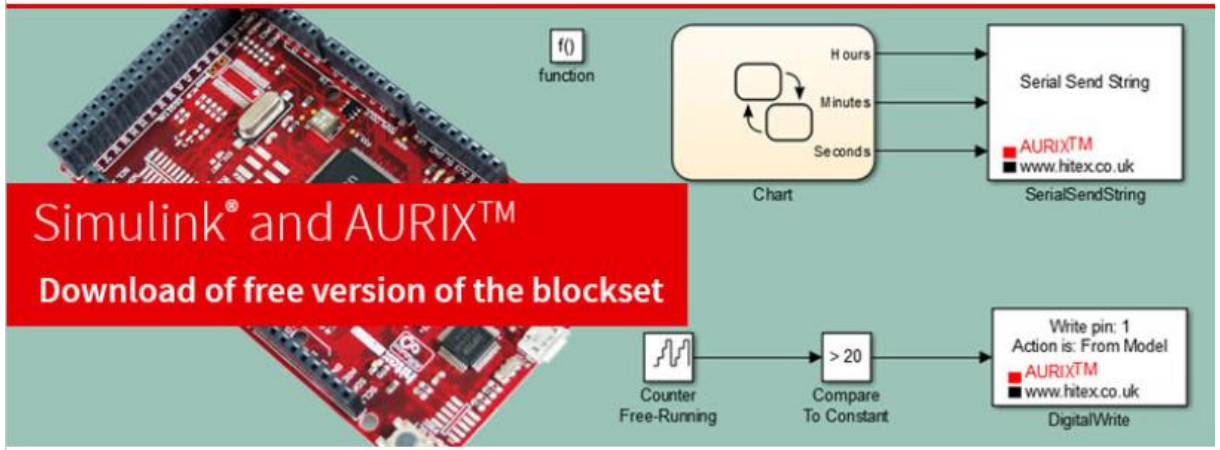
http://www2.hitex.com/l/62422/2018-04-09/4bg88r

제출한 메일 계정으로 다운로드 및 문서 자료 링크 수신

View online

hitex
EMBEDDED TOOLS & SOLUTIONS

Building a safe and secure embedded world



thank you for your interest in our free blockset for Simulink® and AURIX™. Please click on the download links below.

Download AURIX Simulink Free Version

» Download AURIX Simulink Free Version

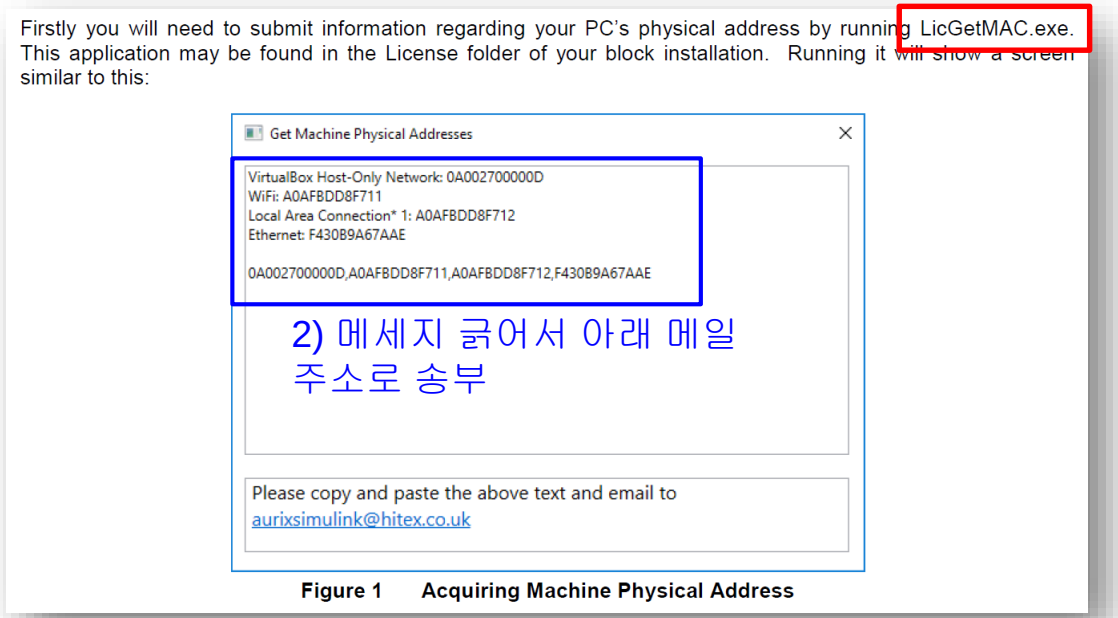
Download AURIX Simulink Presentation

» Download AURIX Simulink Presentation

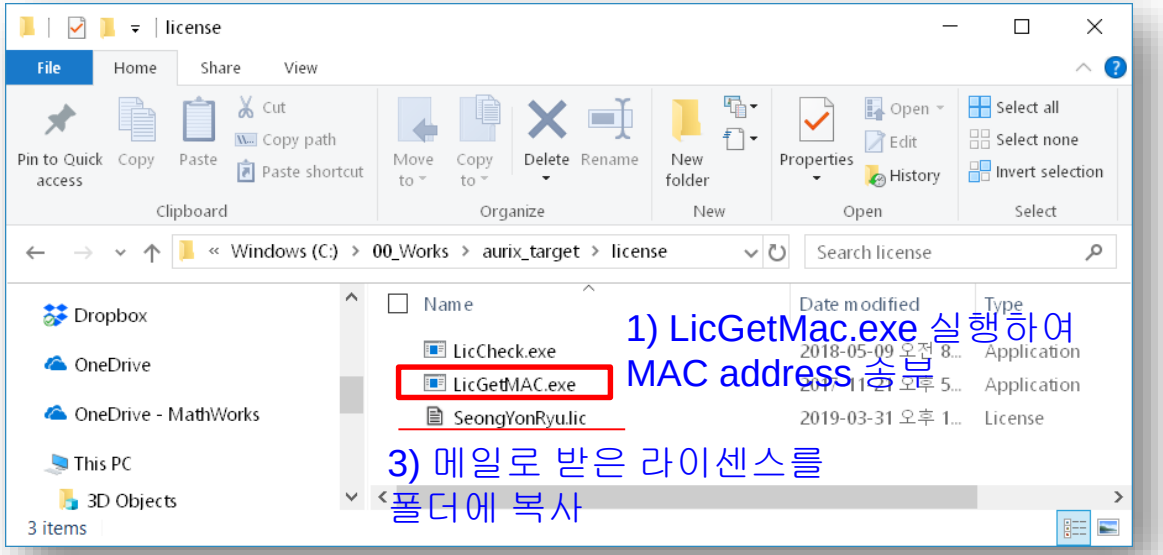
AURIX Simulink Blockset 설치 방법

1) 메일을 통해 받은 zip 파일 압출 풀면 Aurix Simulink blockset 설치 폴더와 매뉴얼(2장 참조)

2) 매뉴얼 2장 가이드라인대로 설치 한 후, MAC address 송부하여 메일로 라이센스 송부 받아야함

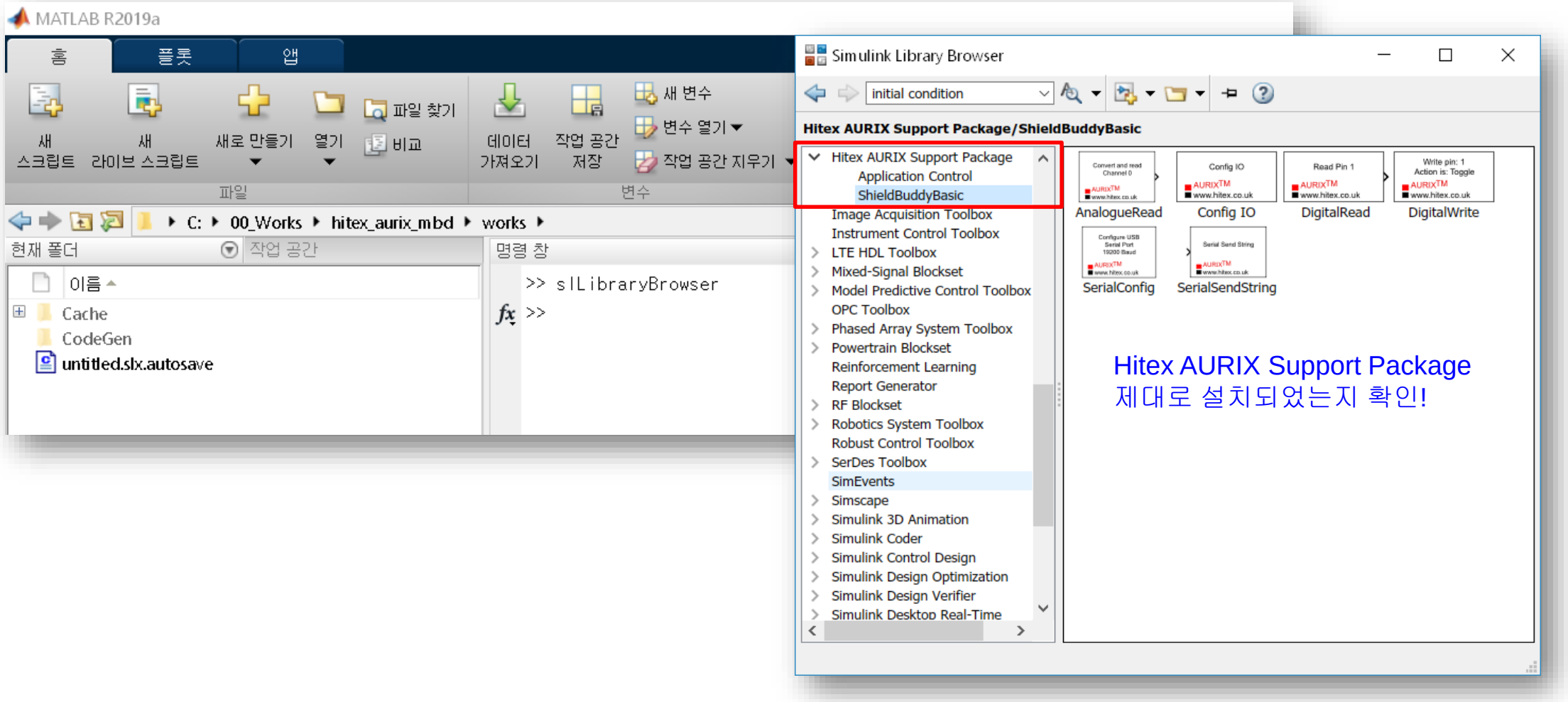


2) 메세지 긁어서 아래 메일 주소로 송부

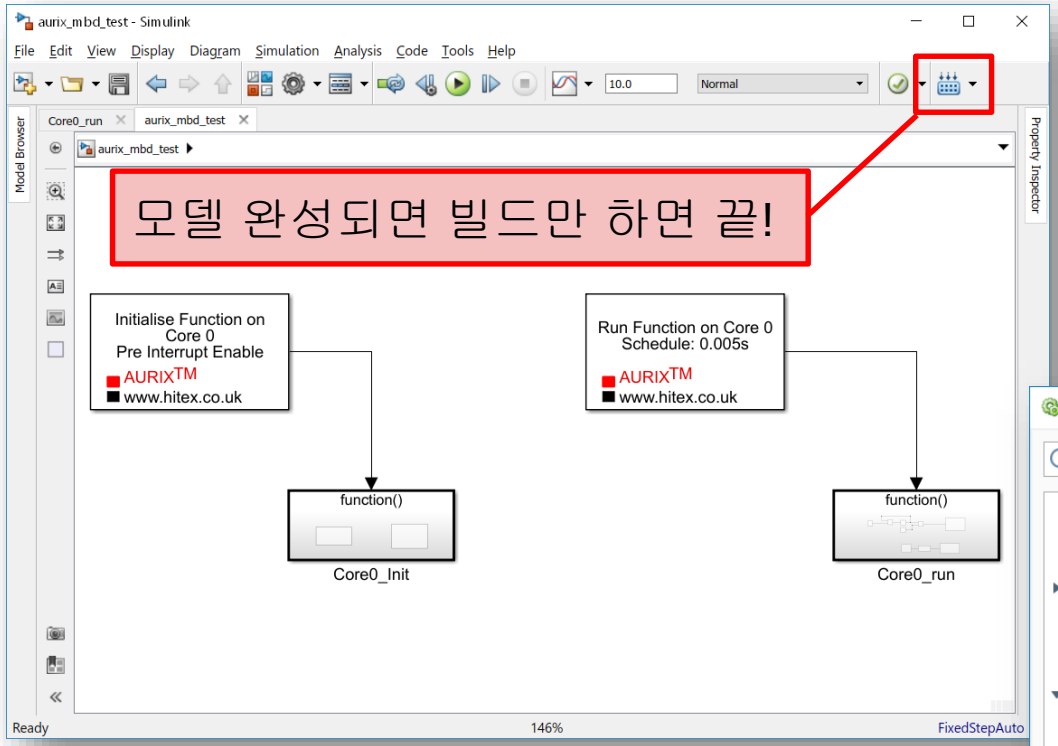


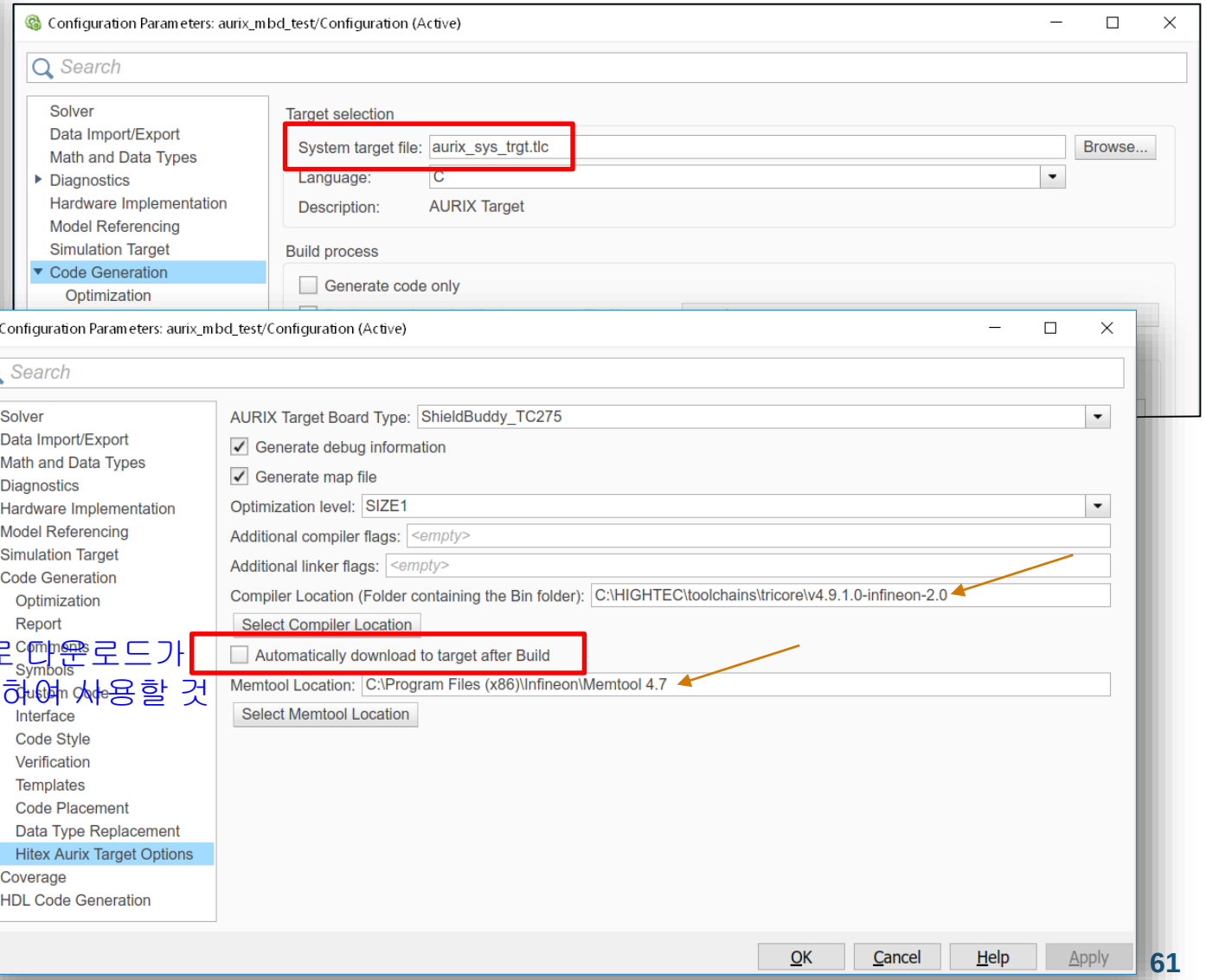
Simulink Library Browser 설치 확인

>> slLibraryBrowser



Demo & Simulink Setting

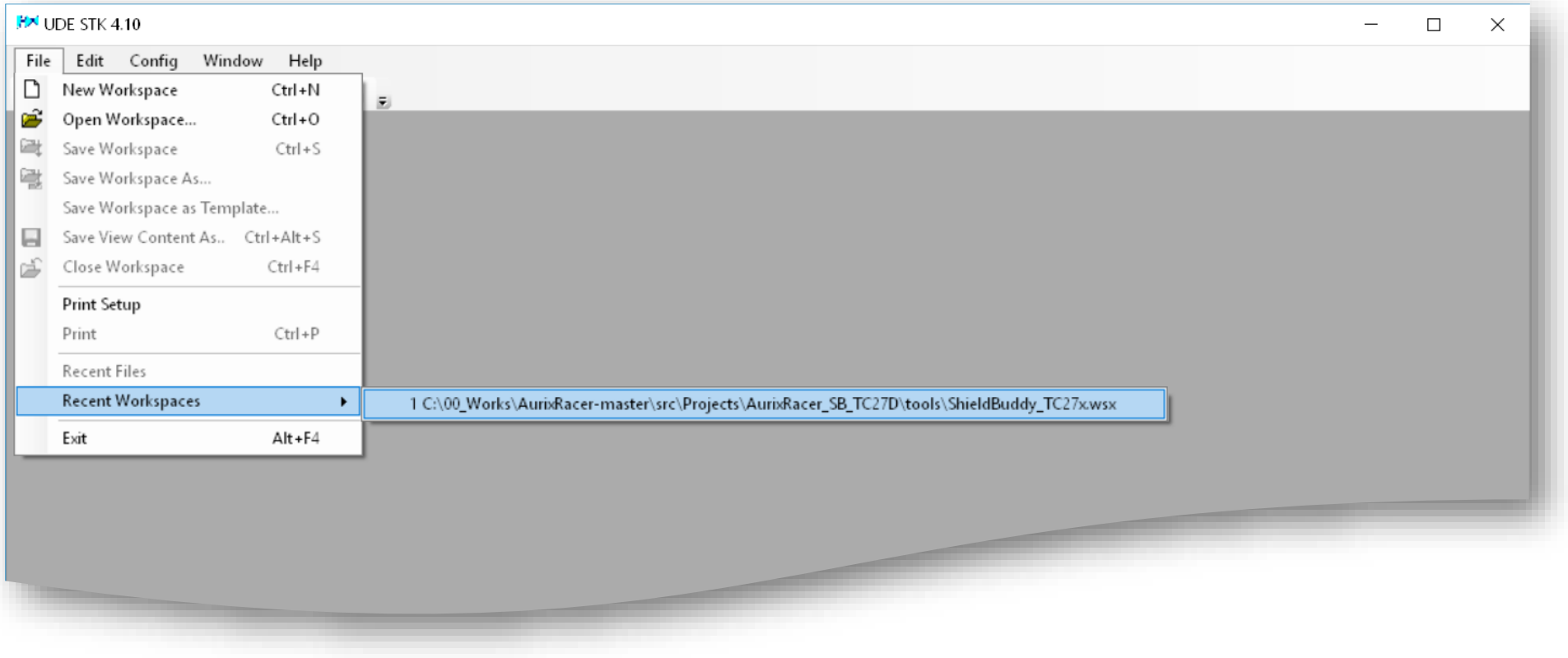




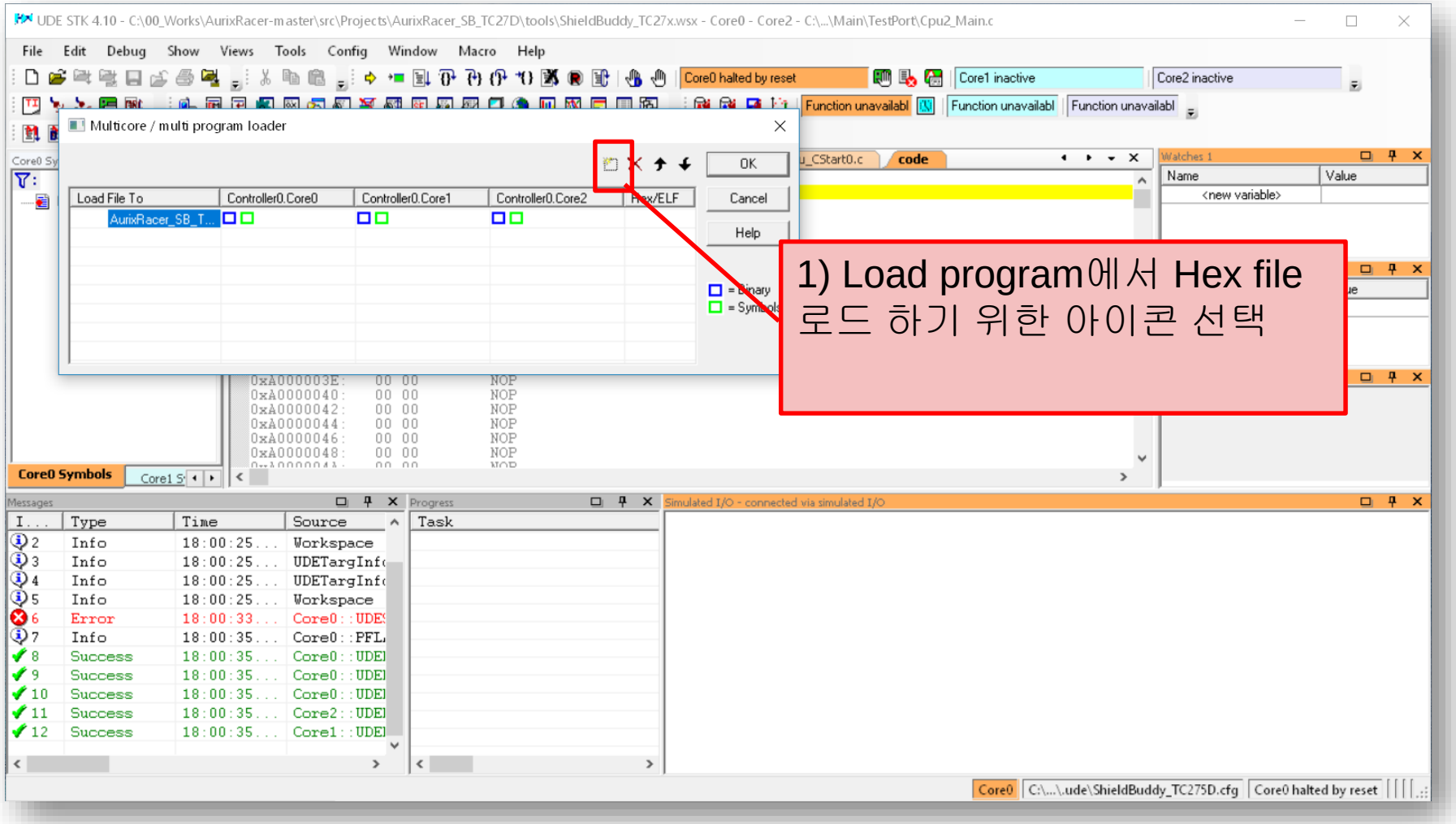
Simulink에서 바로다운로드가
안되면 체크 해제하여 사용할 것

AURIX Simulink Blockset 사용한 경우 Hex 다운로드

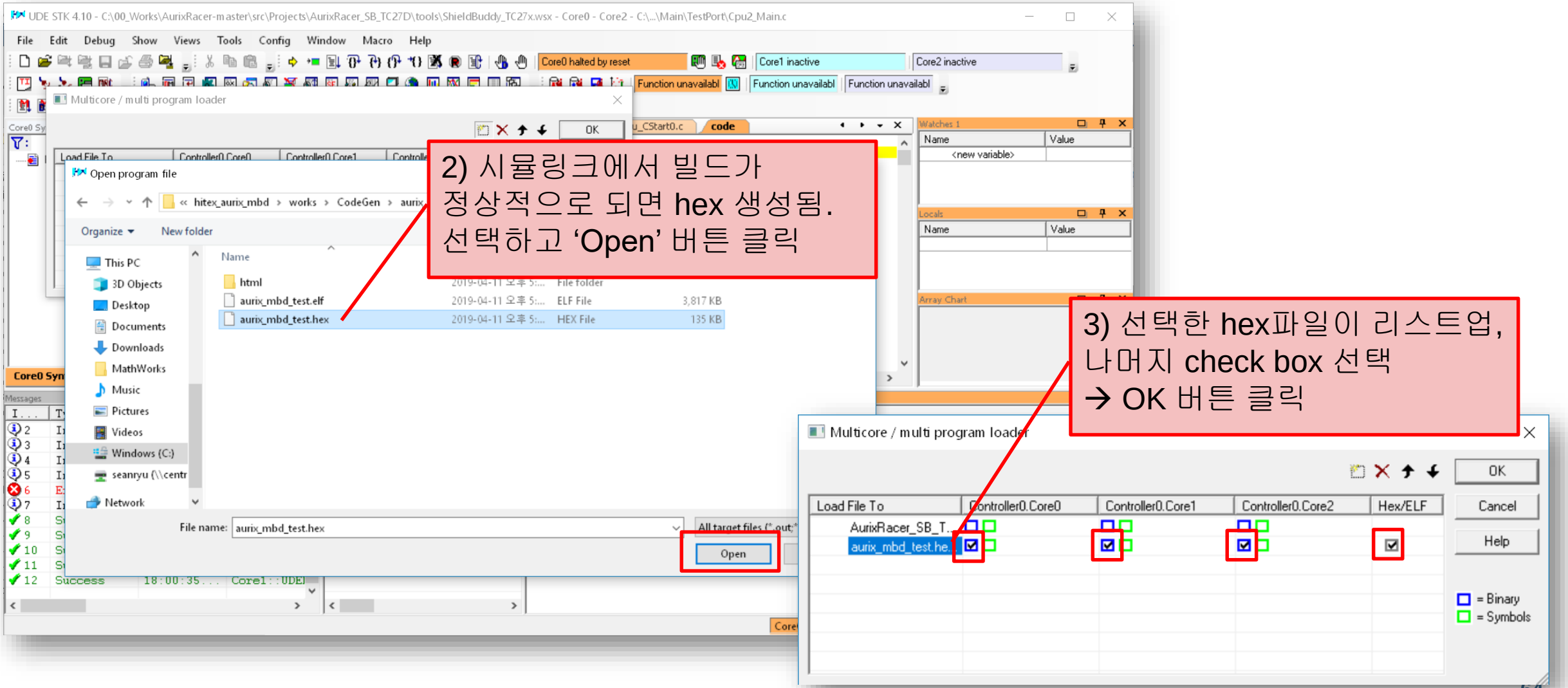
- UDEVisualPlatform.exe 실행
  - AurixRacer 사용위한 세팅 환경 그대로 사용



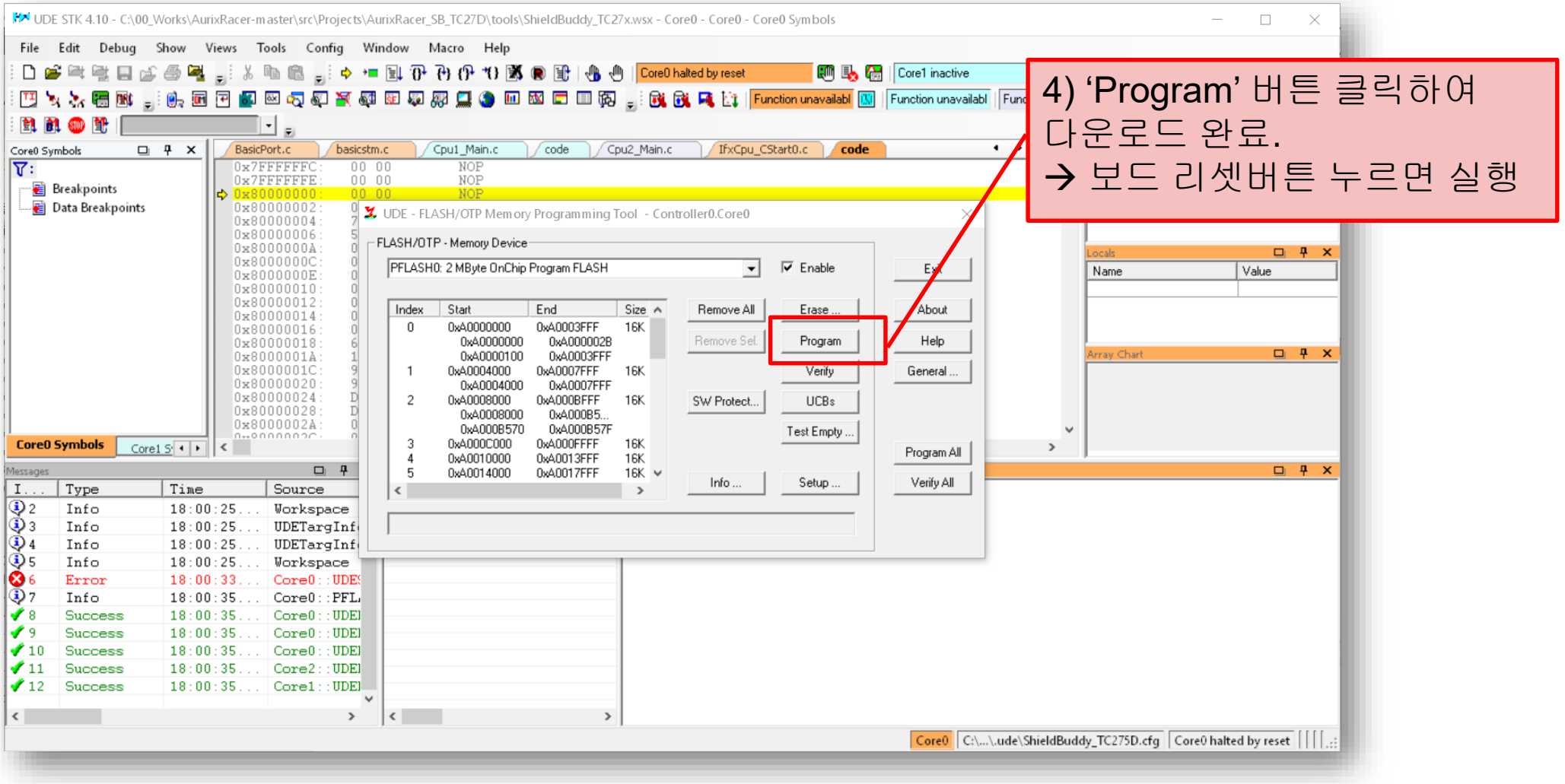
AURIX Simulink Blockset 사용한 경우 Hex 다운로드



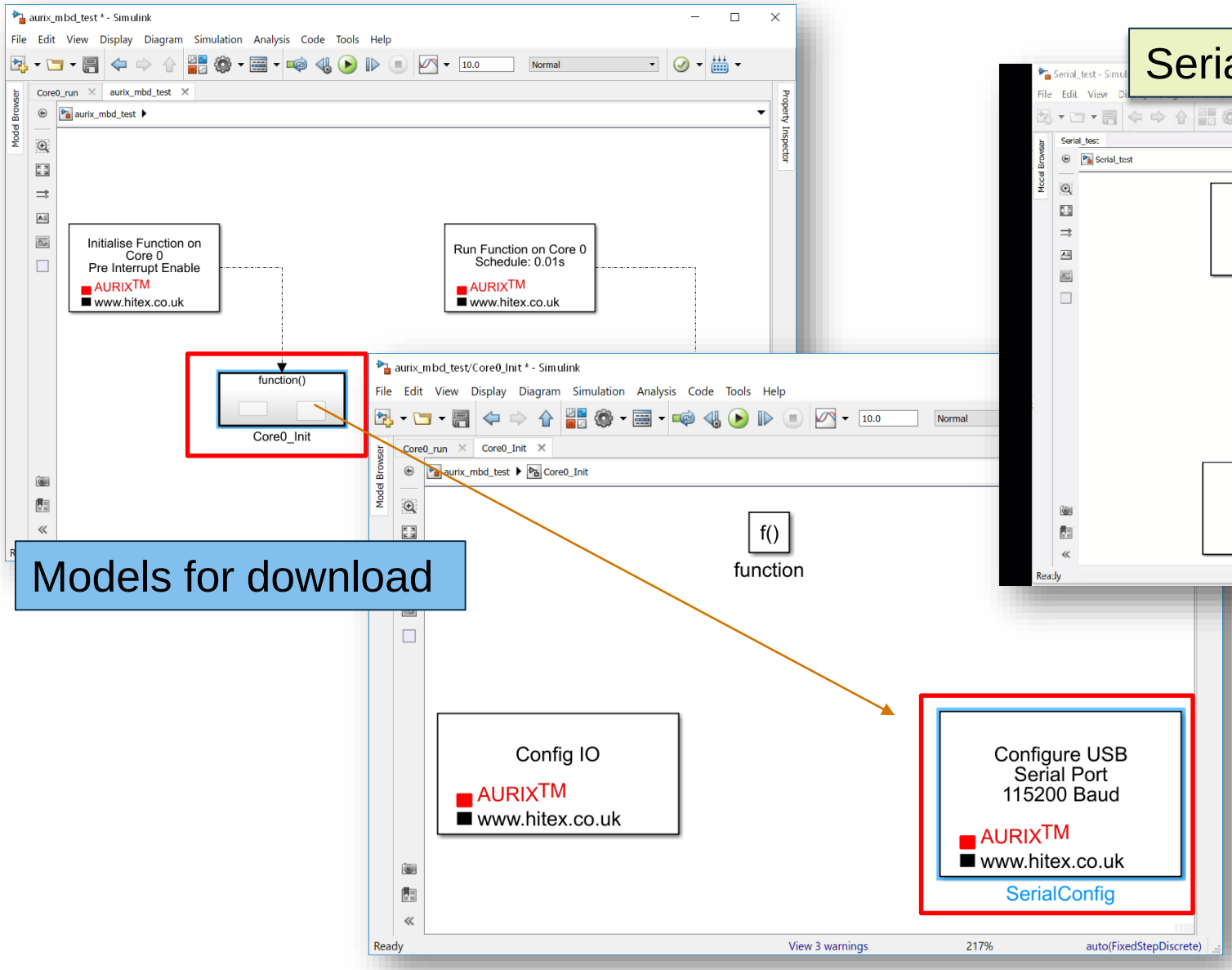
AURIX Simulink Blockset 사용한 경우 Hex 다운로드



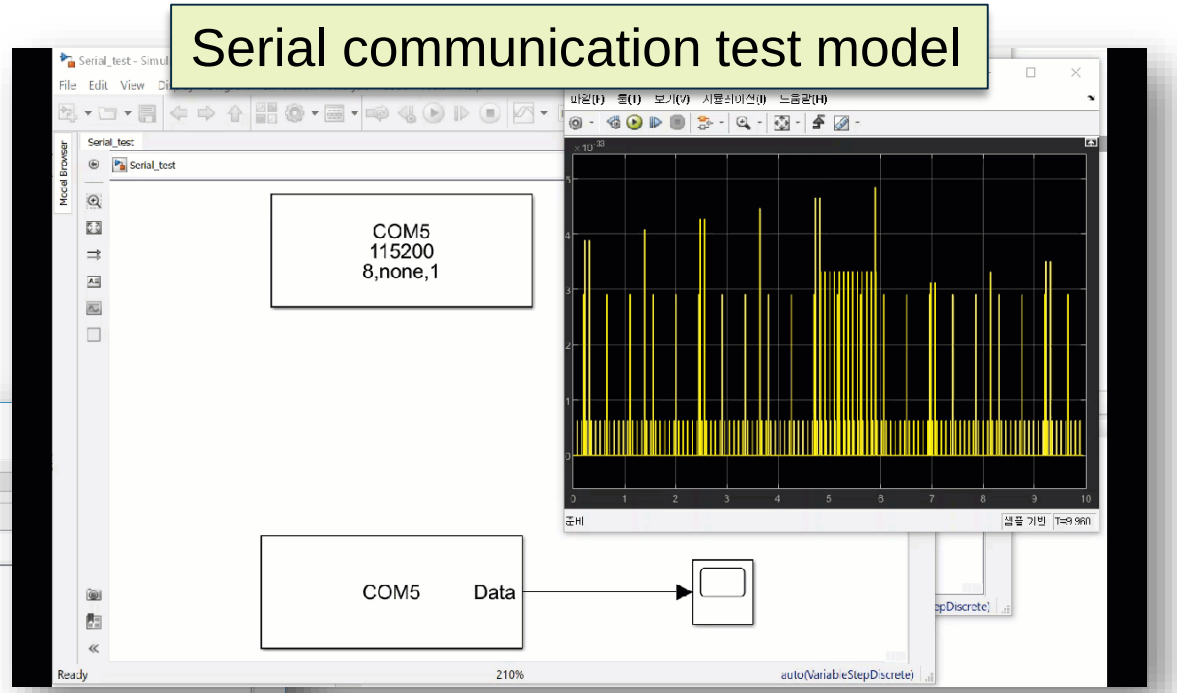
AURIX Simulink Blockset 사용한 경우 Hex 다운로드



Demo: Serial Communication

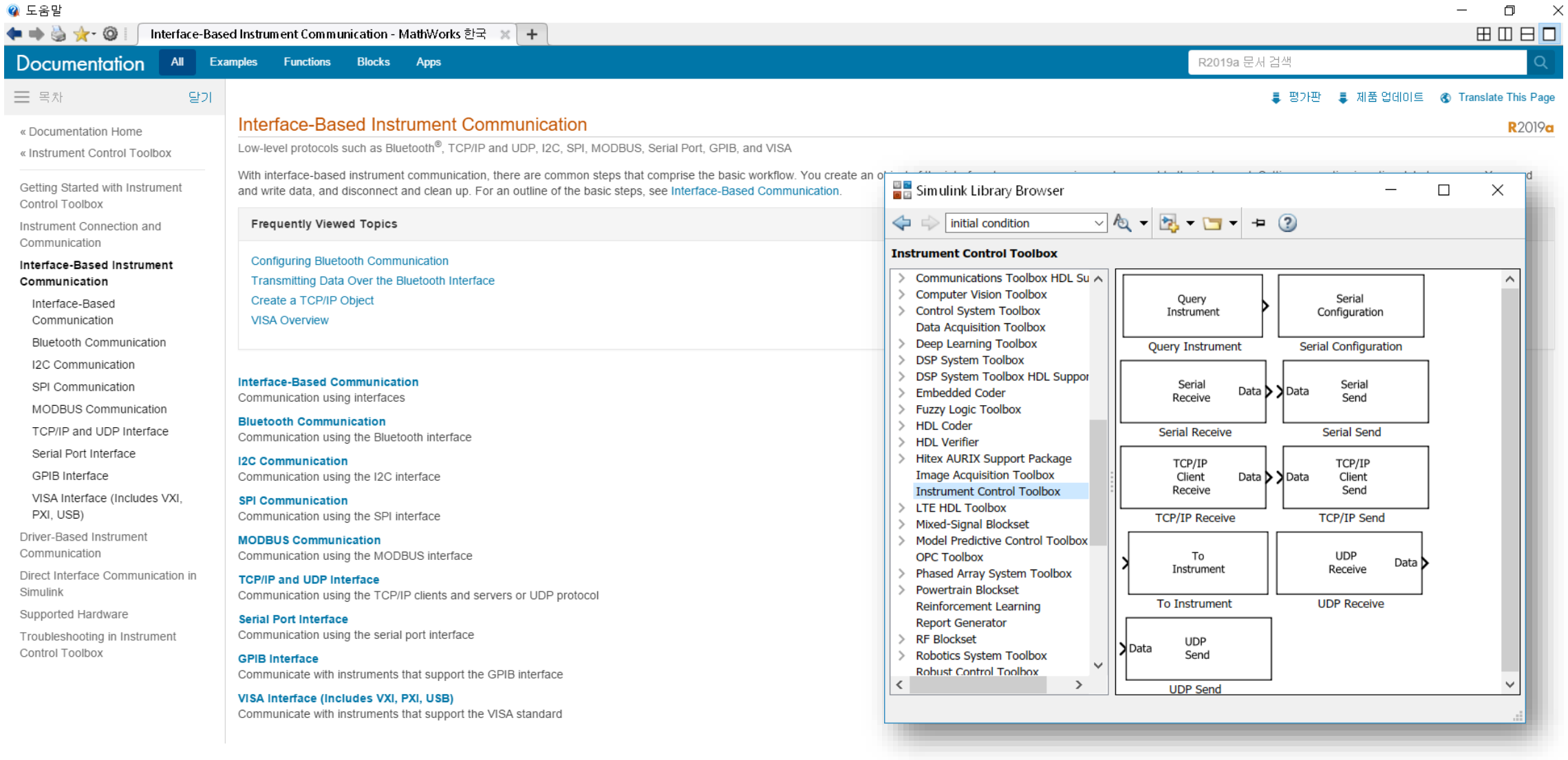


Models for download



Data logging using Instrument Control Toolbox ex. Data acquisition from **line scan camera testing**

Instrument Control Toolbox for Serial Communication



https://www.mathworks.com/help/releases/R2019a/instrument/interface-based-instrument-communication.html

DSP System Toolbox

Design and simulate streaming signal processing systems

DSP System Toolbox™ provides algorithms, apps, and scopes for designing, simulating, and analyzing signal processing systems in MATLAB® and Simulink®. You can model real-time DSP systems for communications, radar, audio, medical devices, IoT, and other applications.

With DSP System Toolbox you can design and analyze FIR, IIR, multirate, multistage, and adaptive filters. You can stream signals from variables, data files, and network devices for system development and verification. The Time Scope, Spectrum Analyzer, and Logic Analyzer let you dynamically visualize and measure streaming signals. For desktop prototyping and deployment to embedded processors, including ARM® Cortex® architectures, the toolbox supports C/C++ code generation. It also supports bit-accurate fixed-point modeling and HDL code generation from filters, FFT, IFFT, and other algorithms.

Algorithms are available as MATLAB functions, System objects, and Simulink blocks.

Getting Started
Learn the basics of DSP System Toolbox

Signal Generation, Manipulation, and Analysis
Create, import, export, display, and manage signals

Filter Design and Analysis
FIR, IIR, frequency transformations

Filter Implementation
Single-rate, multirate, and adaptive filters

Transforms and Spectral Analysis
FFT, DCT, spectral analysis, linear prediction

Statistics and Linear Algebra
Measurements, statistics, matrix math, linear algebra

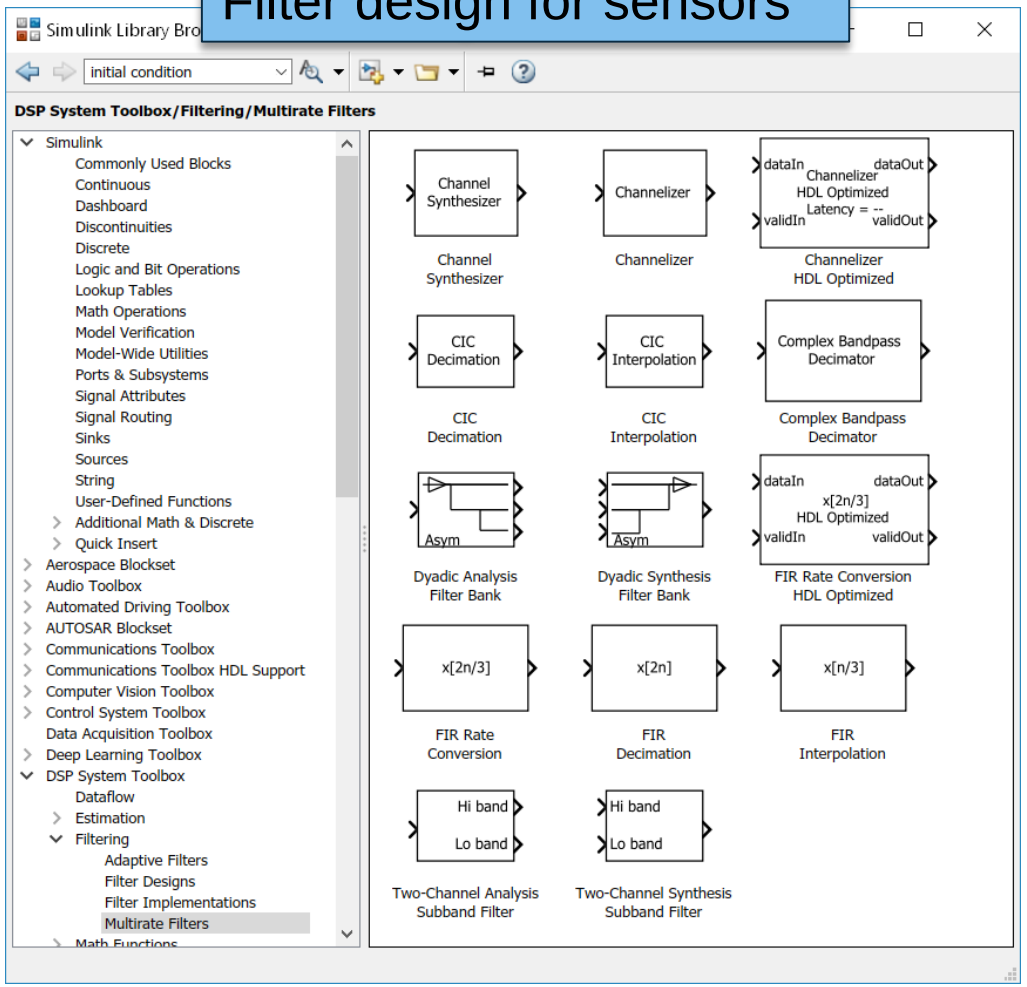
Fixed-Point Design
Floating-point to fixed-point conversion, fixed-point algorithm design.

Code Generation
Simulation acceleration, code generation, optimization for ARM Cortex-M Processors and ARM Cortex-A Processors

Applications
Simulate radar, communication, and biomedical systems

Supported Hardware
Support for third-party hardware

Filter design for sensors

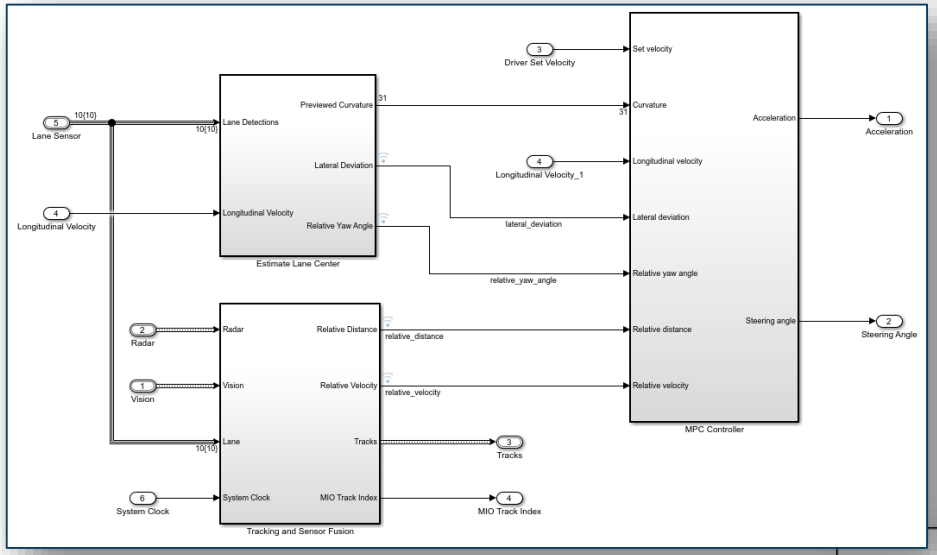


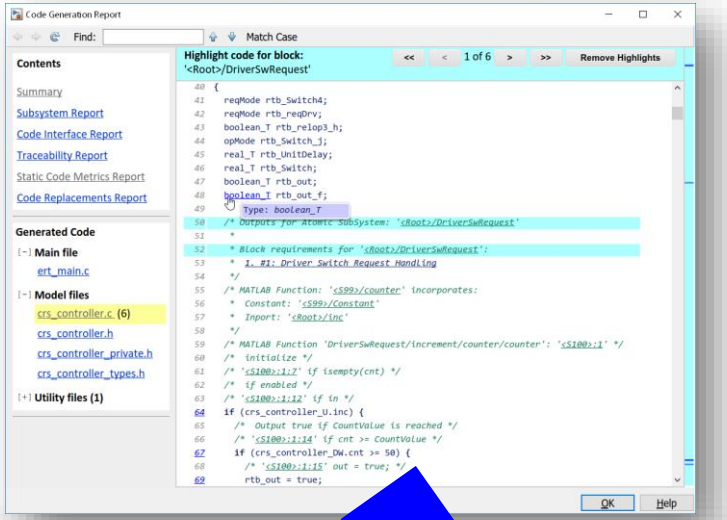
https://www.mathworks.com/help/releases/R2019a/dsp/index.html

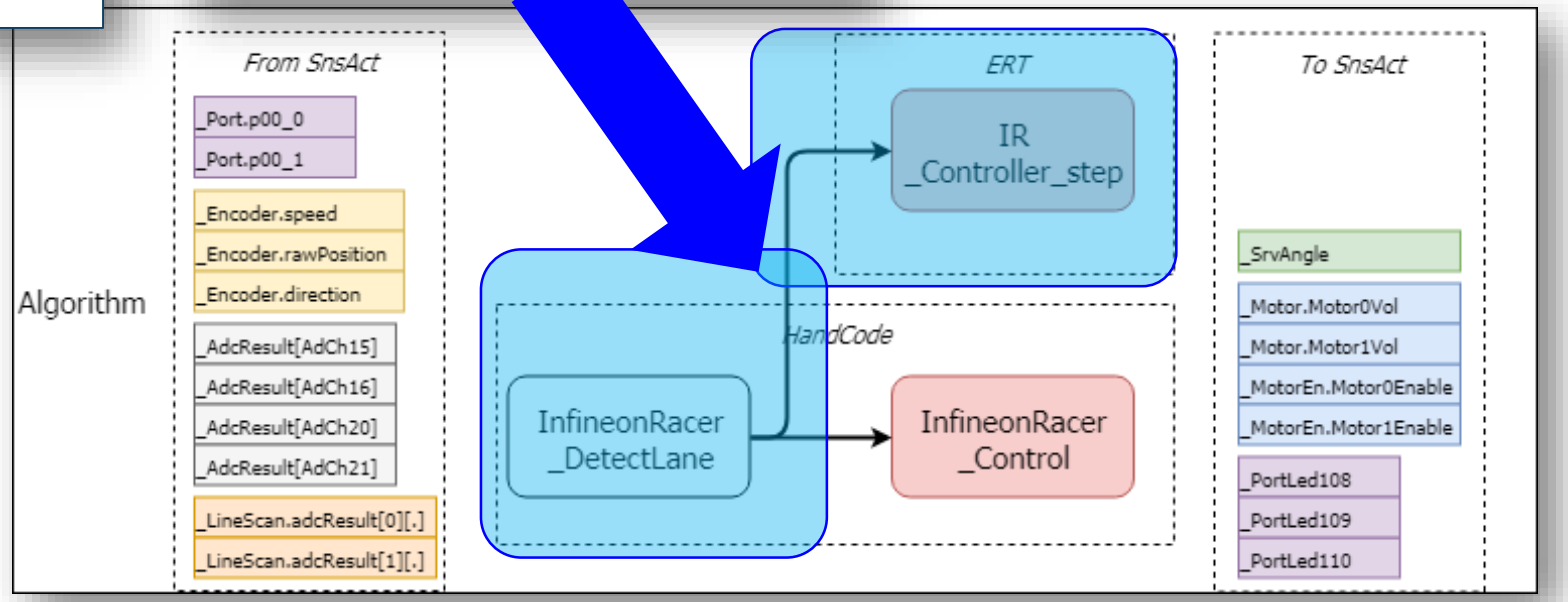
Getting Started from Simulink

- AurixRacer Platform

Remind from Software Integration…

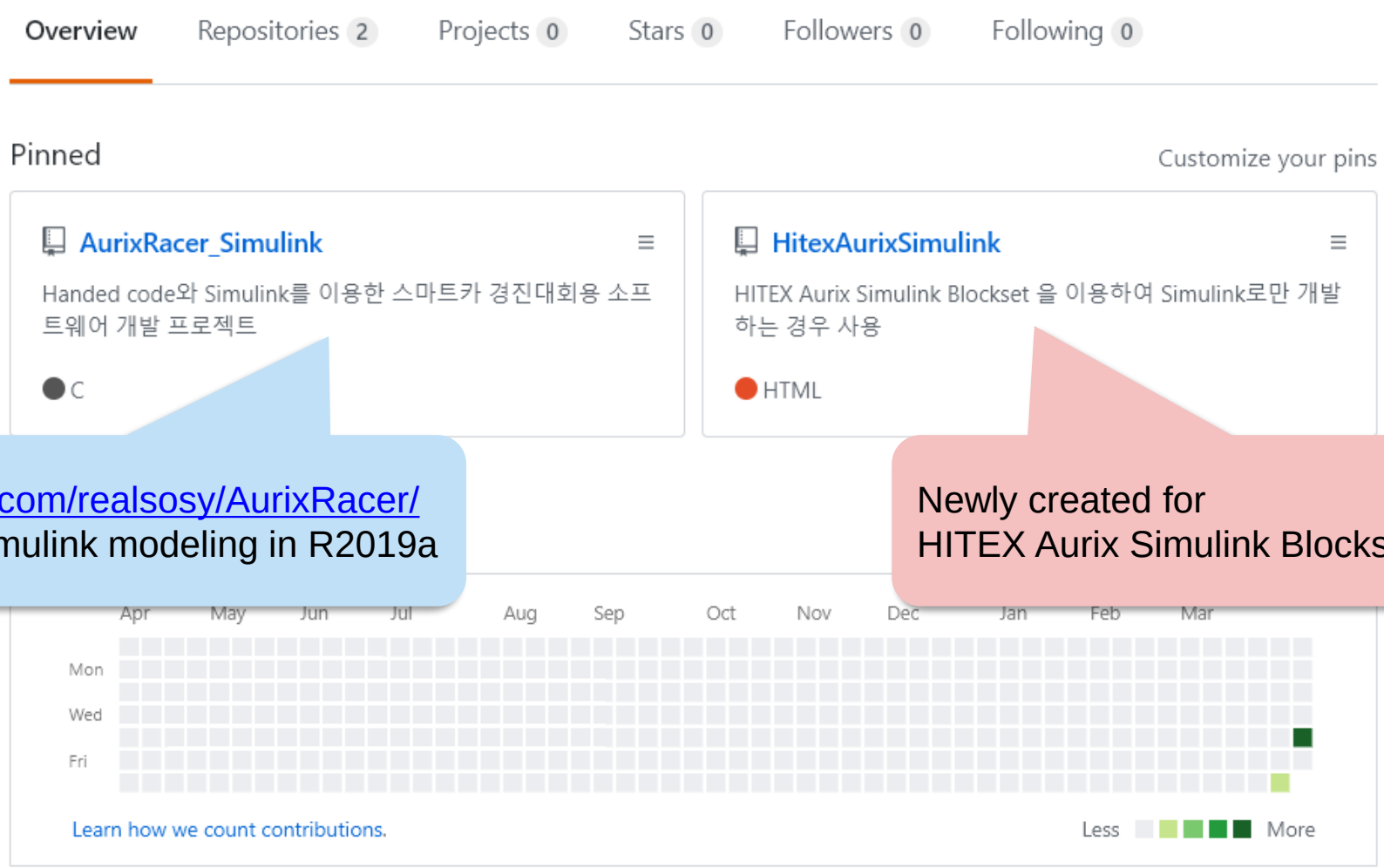






https://aurixracer.readthedocs.io/ko/latest/SimulinkAsAProgrammingLanguage/index.html

Simulink Resource for SMCC



From https://github.com/realsosy/AurixRacer/
And modified for Simulink modeling in R2019a

Newly created for
HITEX Aurix Simulink Blockset

https://github.com/seongyon

Successful collaborative development requires…

1. Same source code, tests, doc, requirements, …

.m .slx

Successful collaborative development requires…

1. Same source code, tests, doc, requirements, …

2. Integration with source control

Successful collaborative development requires…

1. Same source code, tests, doc, requirements, …

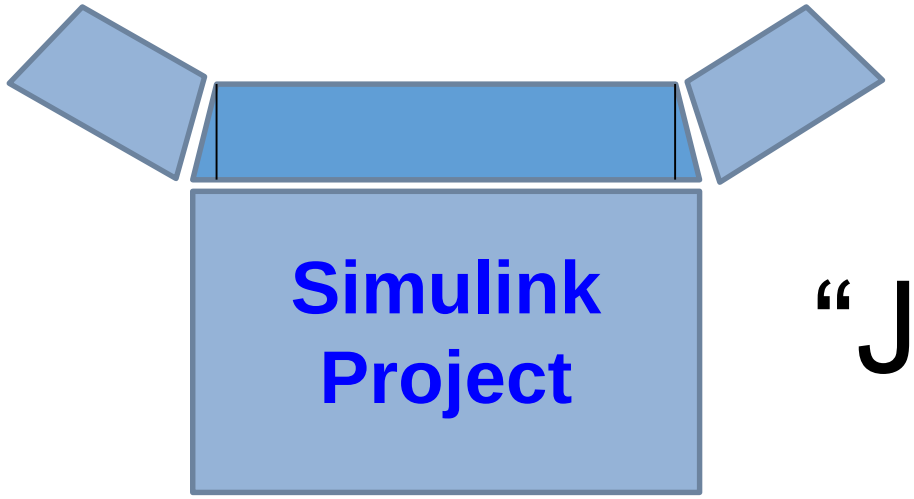
2. Integration with source control

3. Consistent, shared environment

MATLAB Path | Workspace Data | C Compiler | Apps & Toolboxes | …

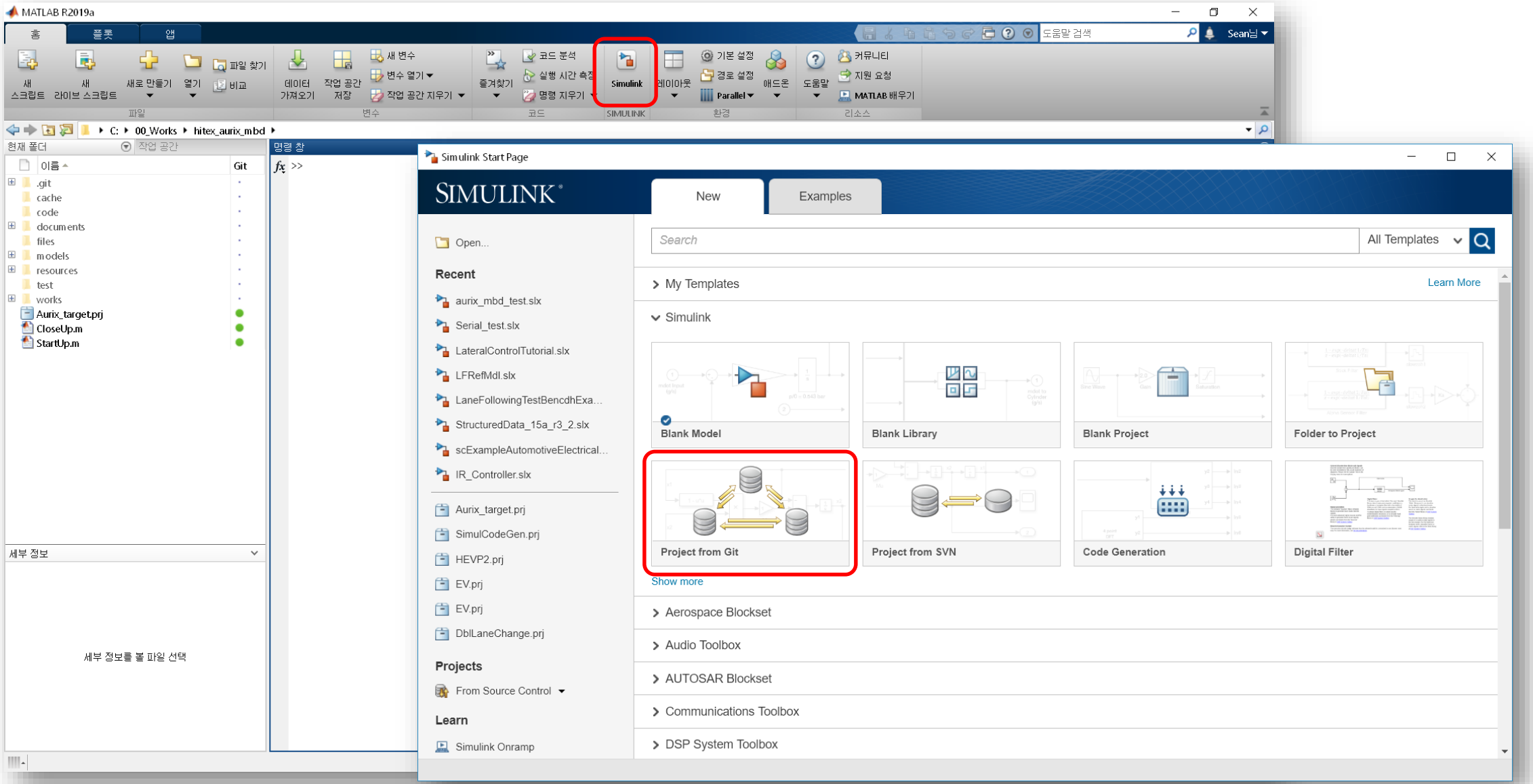
Successful collaborative development requires…

1. Same source code, tests, doc, requirements, …
2. Integration with source control
3. Consistent, shared environment

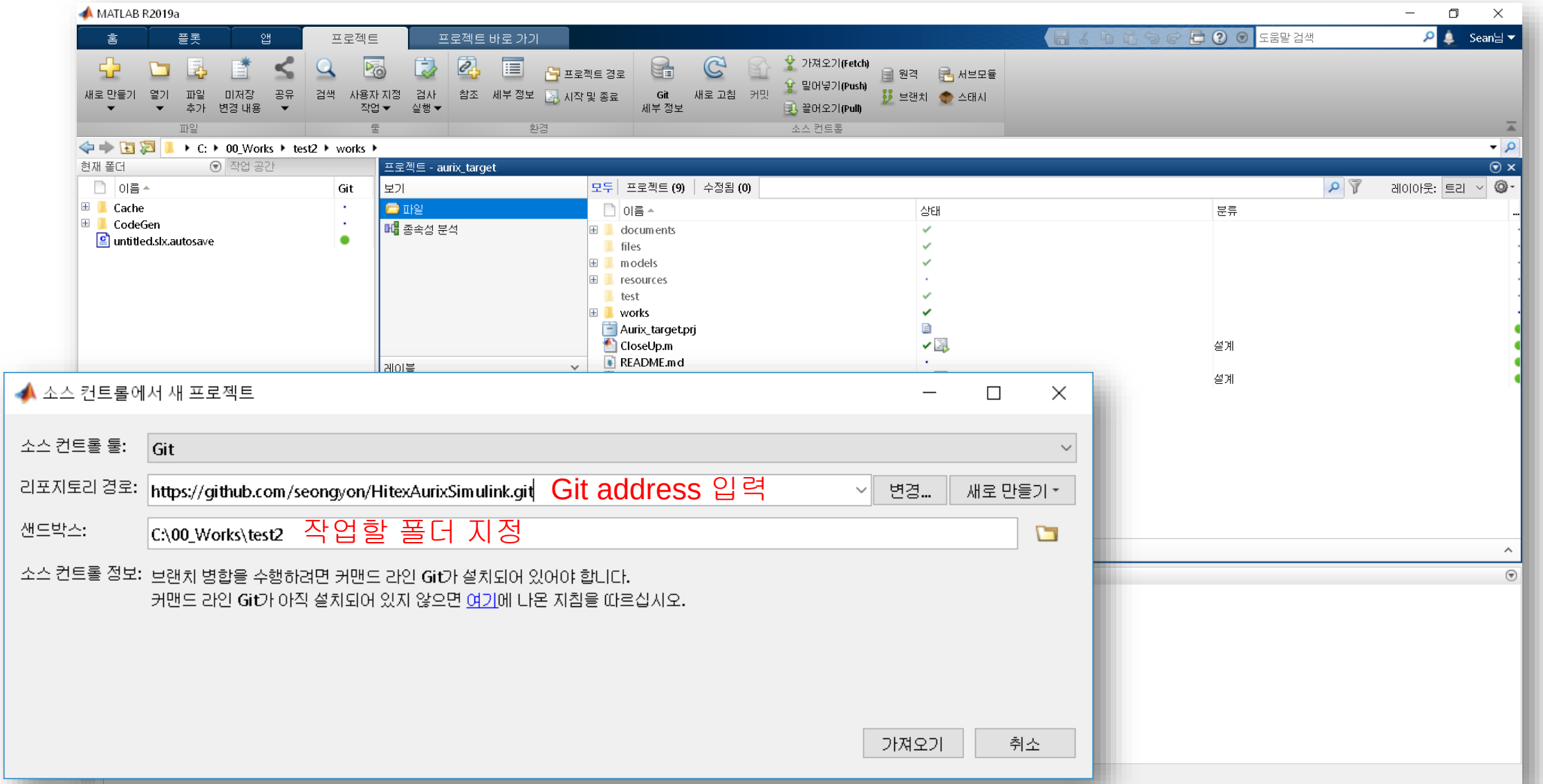


"Just works"

Let's Practice…



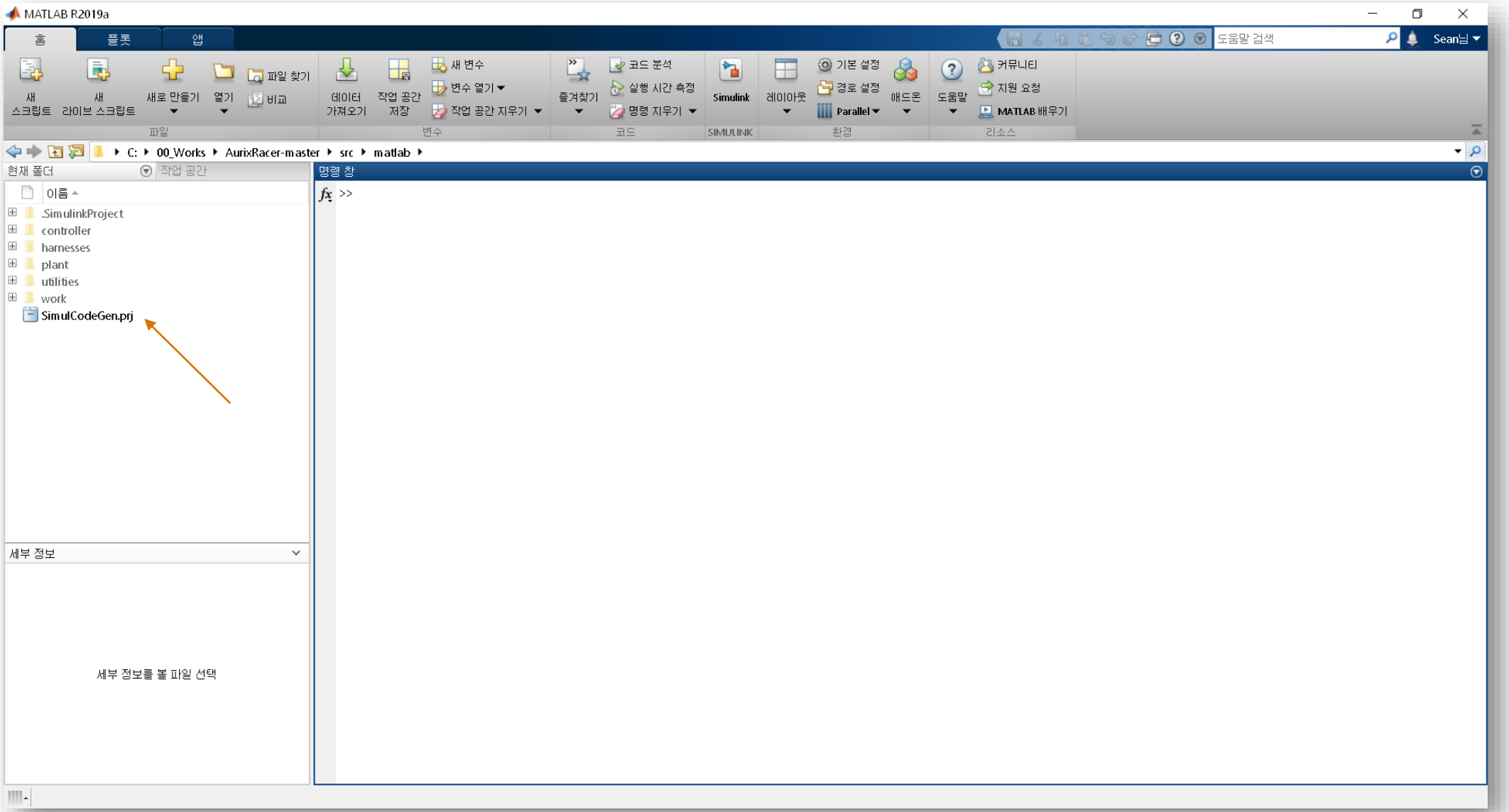
Project from Git: HITEX Aurix Simulink Blockset



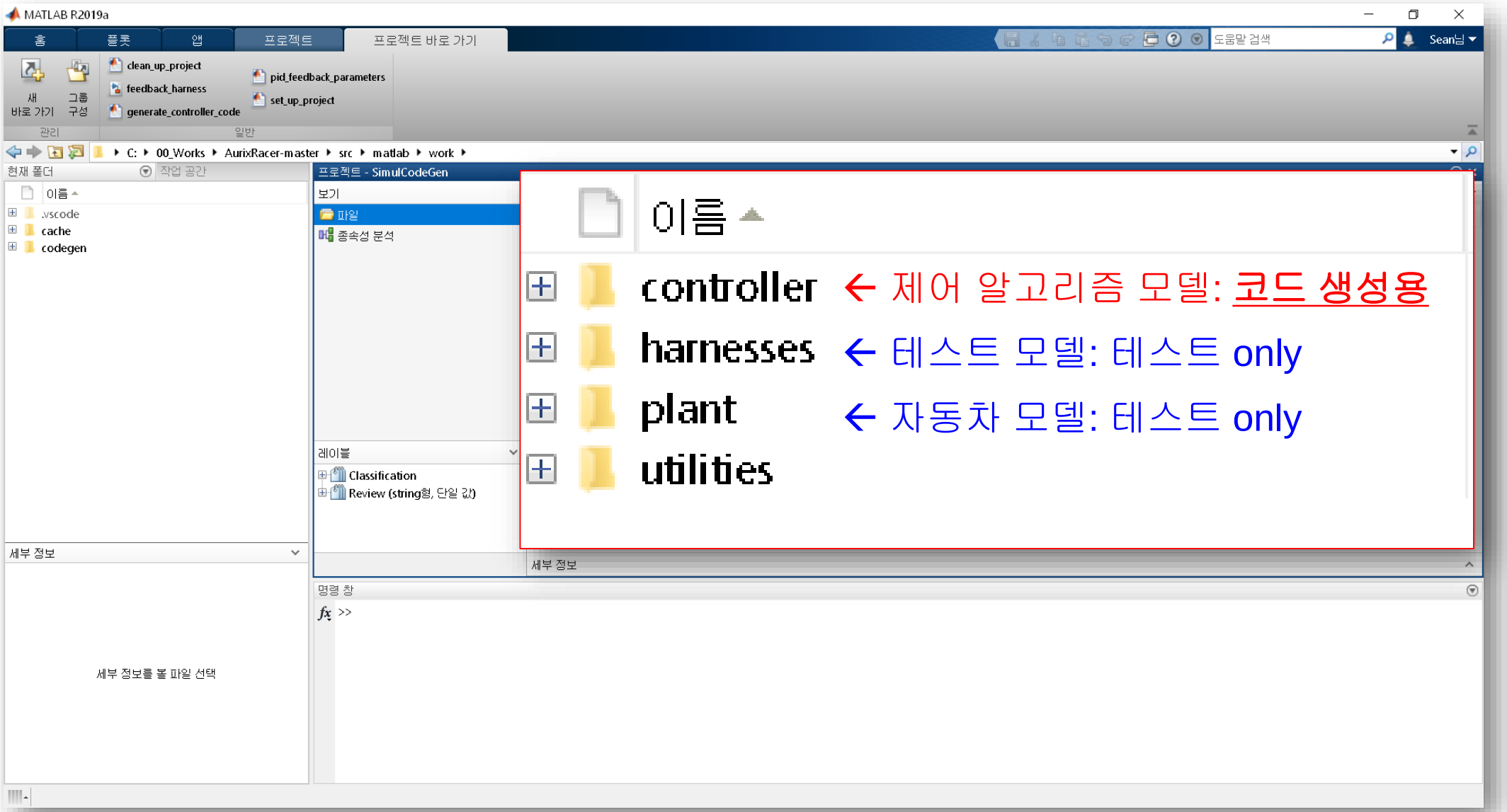
https://github.com/seongyon/HitexAurixSimulink.git

AurixRacer for Simulink

Starting Simulink Project



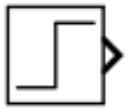
Folder Structure

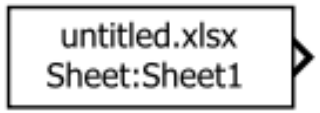


How to Test Your Model…?

Signal Builder

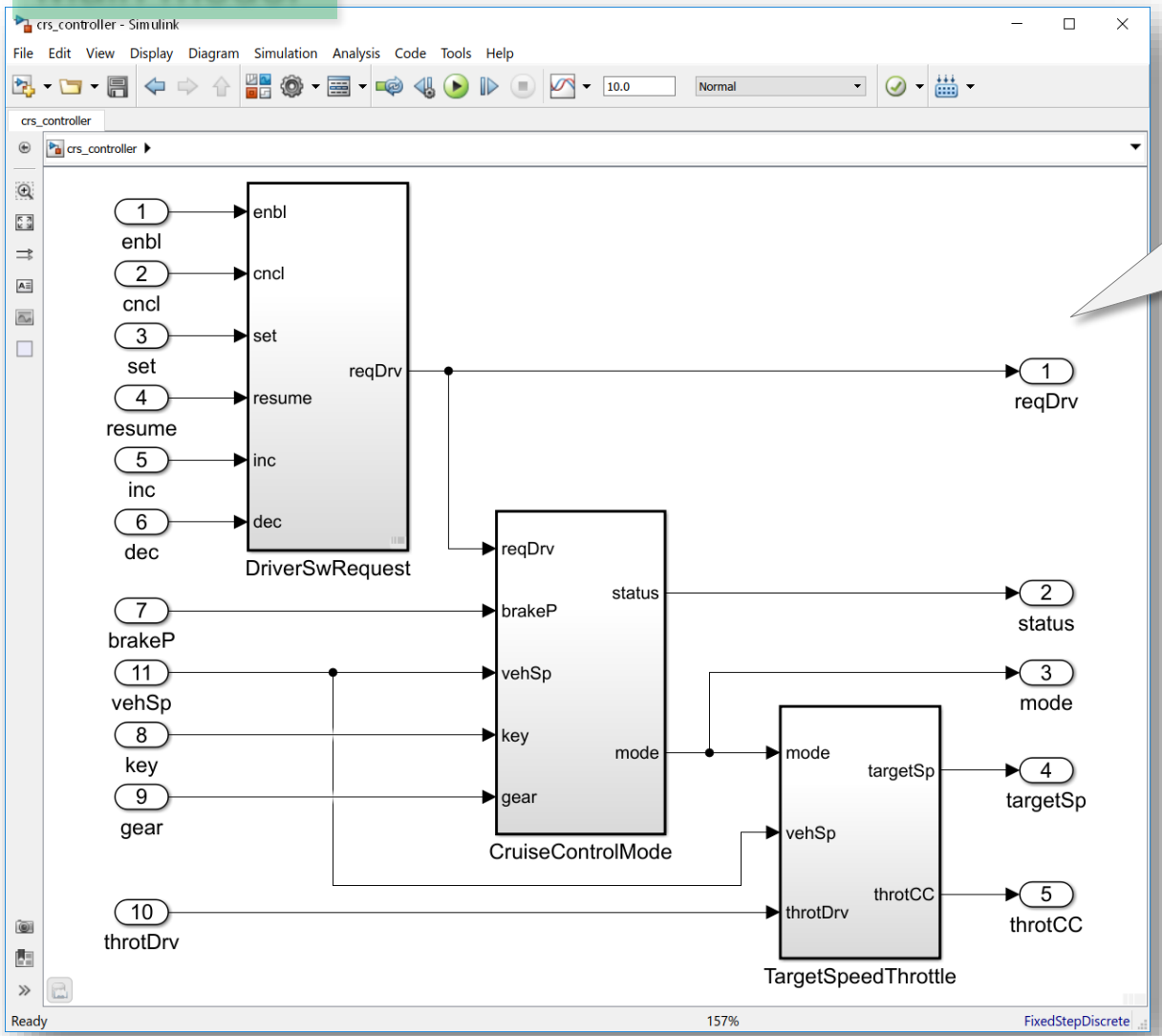
Signal Editor


Step



From Spreadsheet

Main model

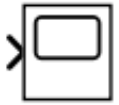


Do you need to modify it for **testing**?


Gauge


Rotary Switch

Slider

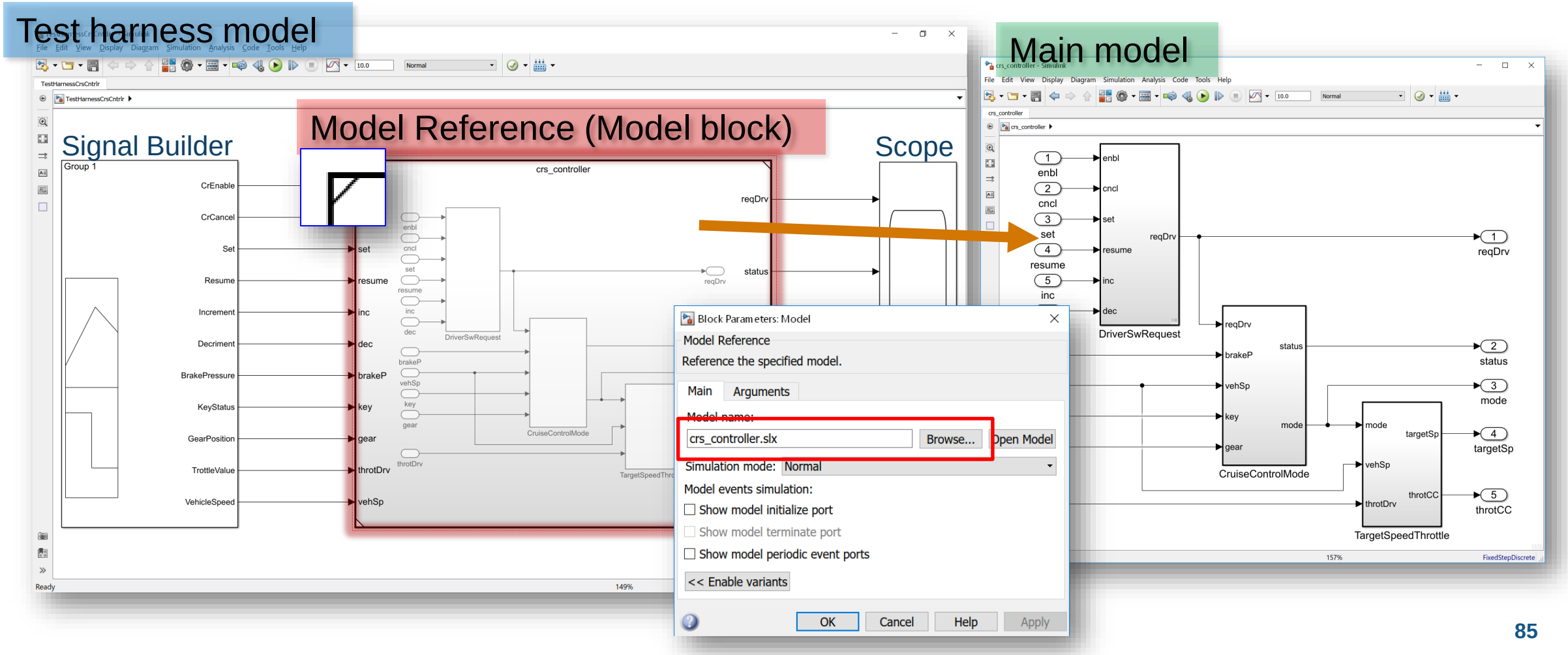

Scope


Test Condition

Verification Subsystem

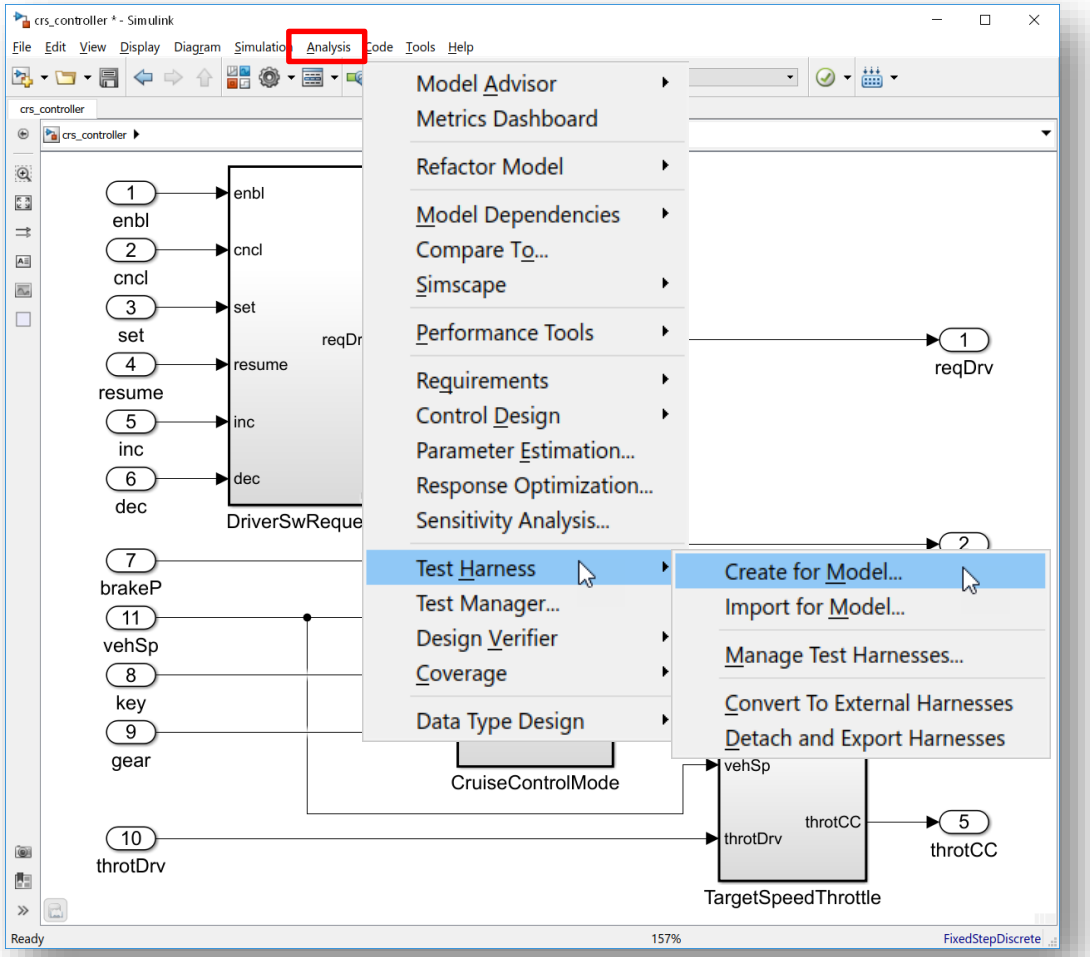
Building Test Harness Model using Model Reference

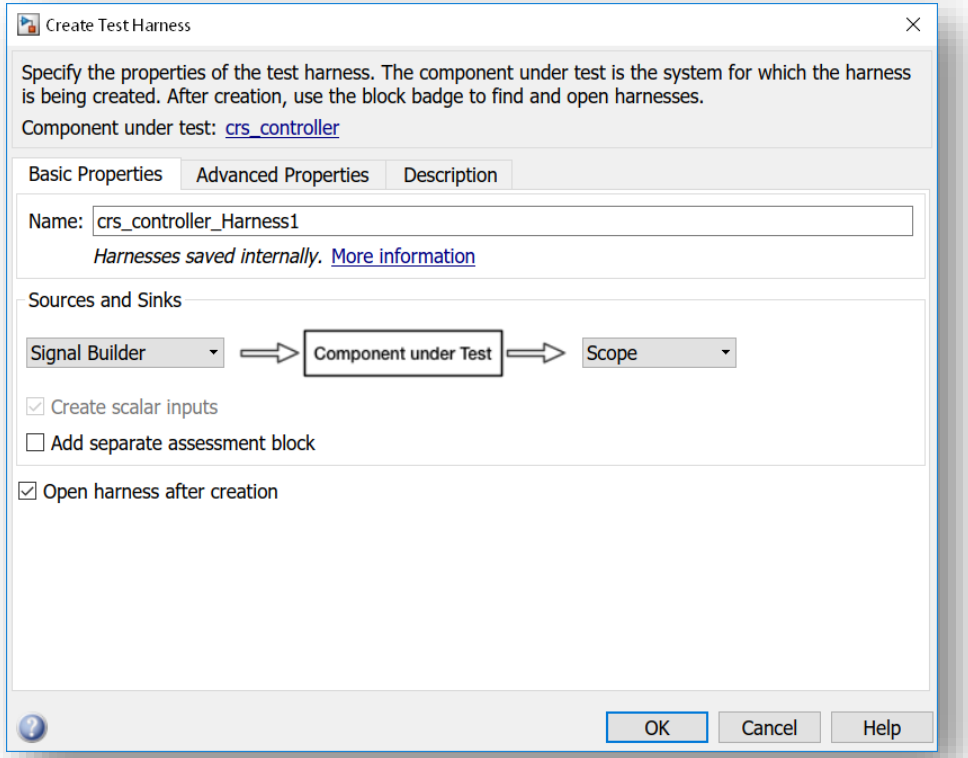
- Separated model not for code generation but only for testing



Test Harness Model

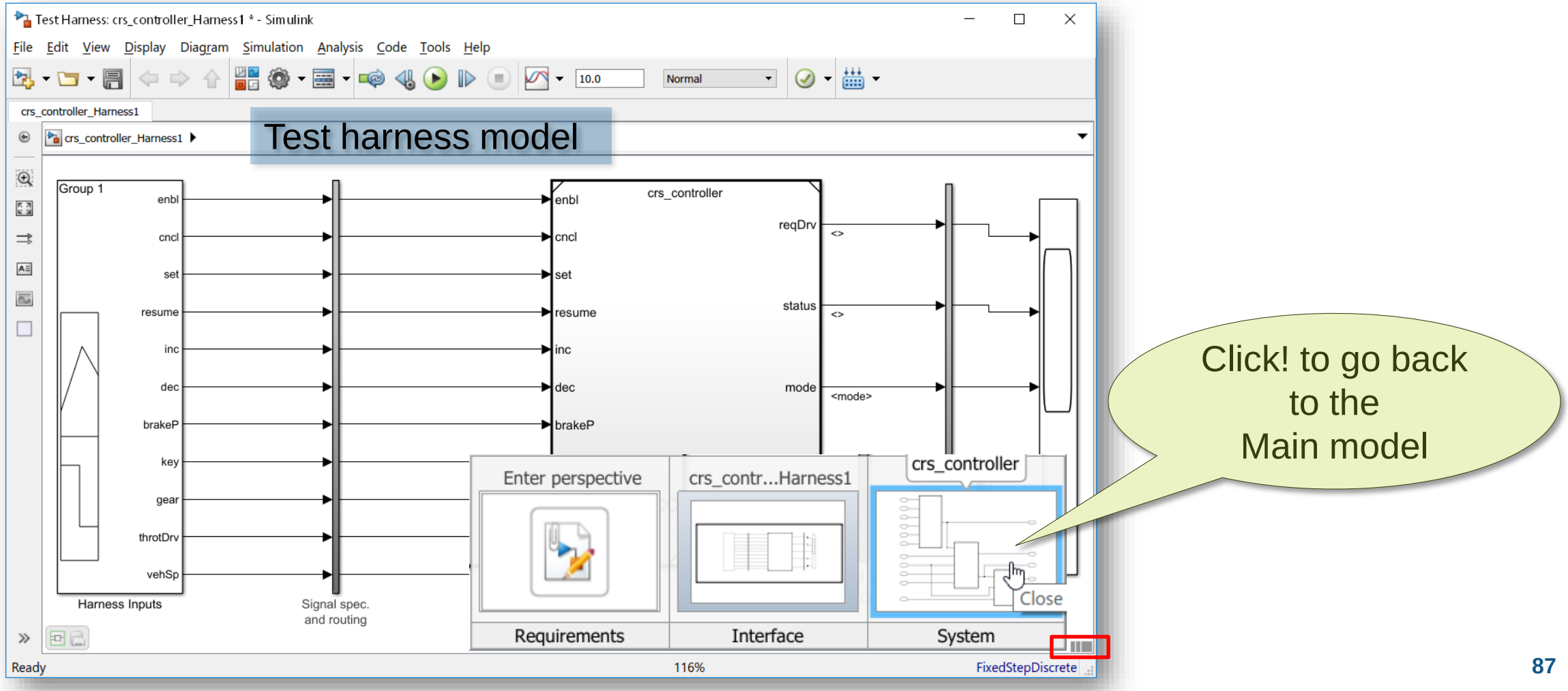
- Test harness model can be included in you original models





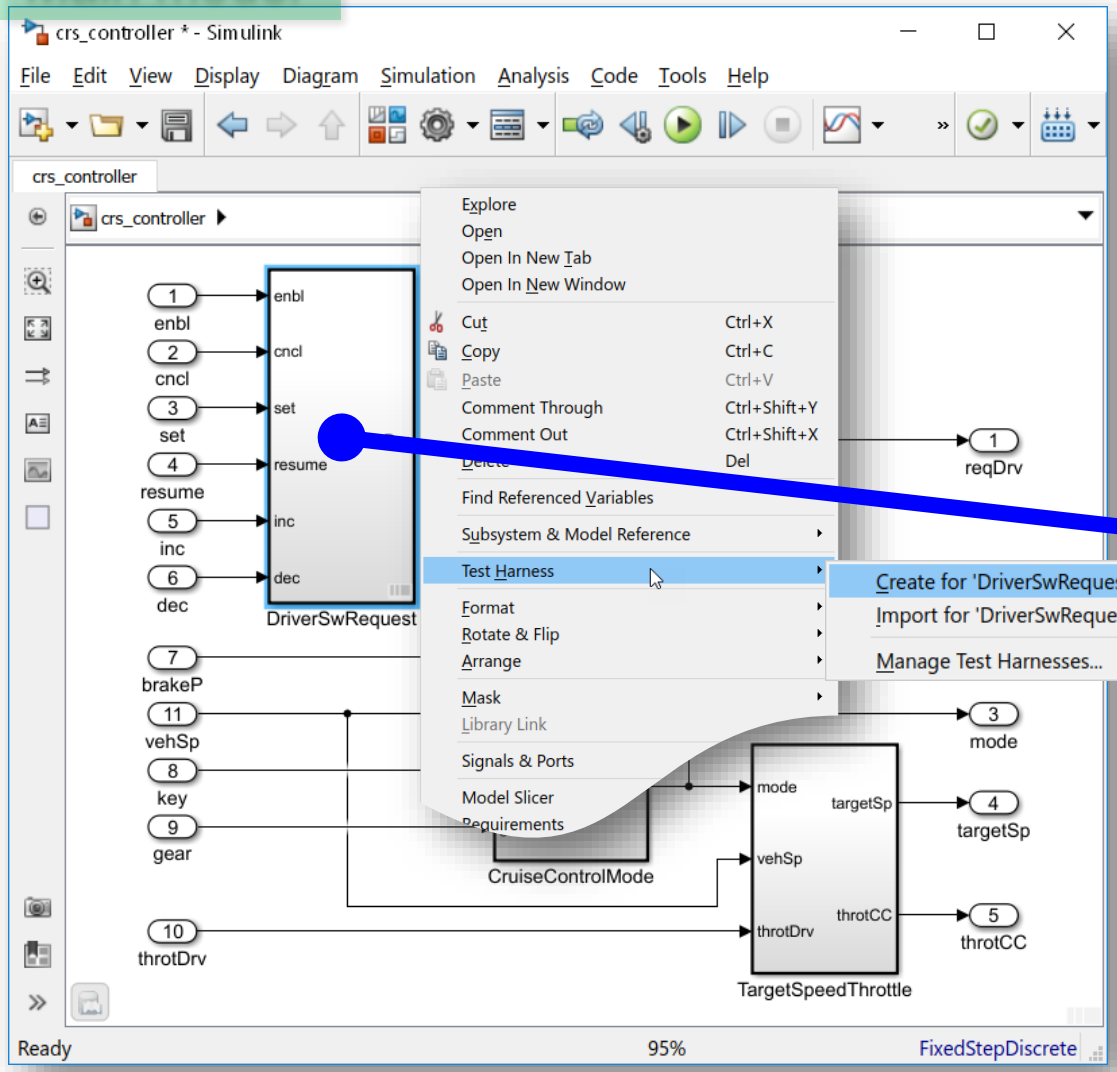
Test Harness Model

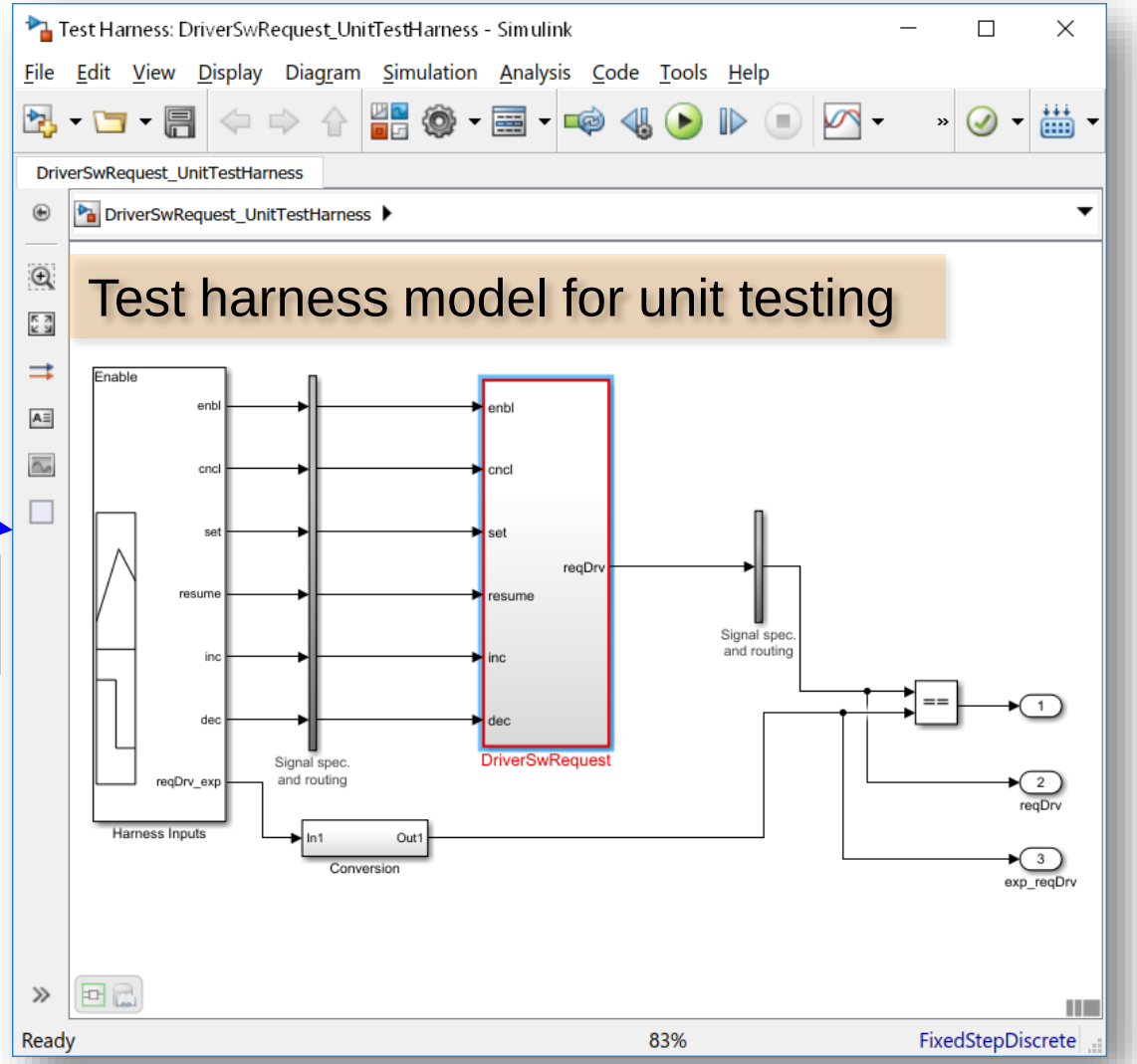
- Test harness model can be included in you original models



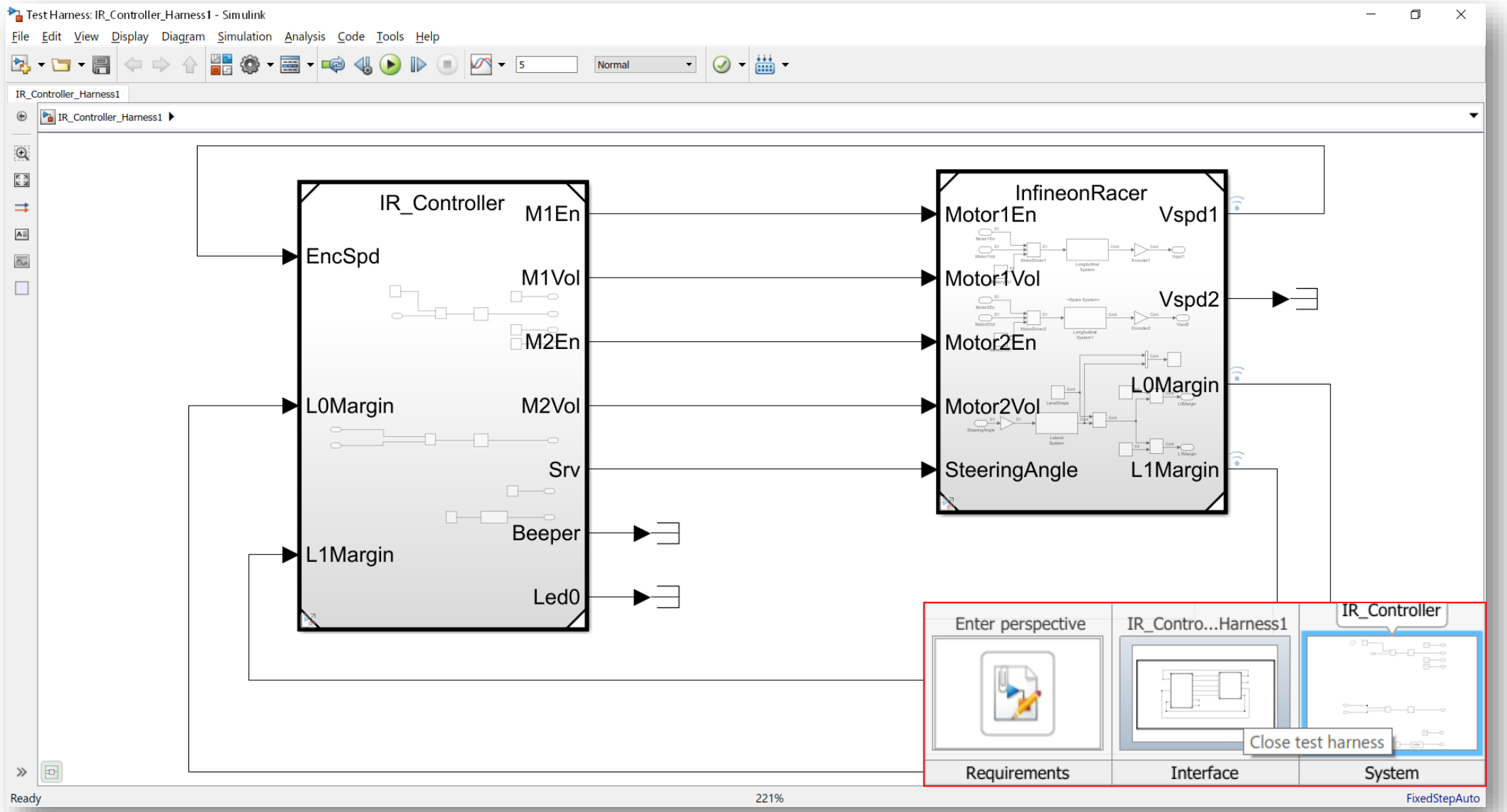
Unit Test using Test Harness Model

Main model



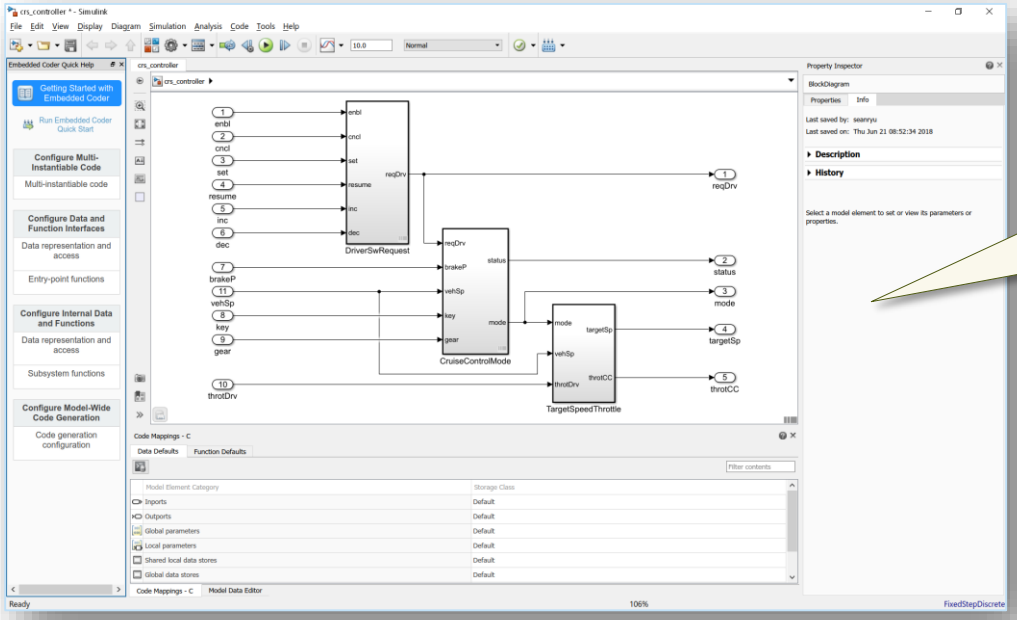


Let's Practice…

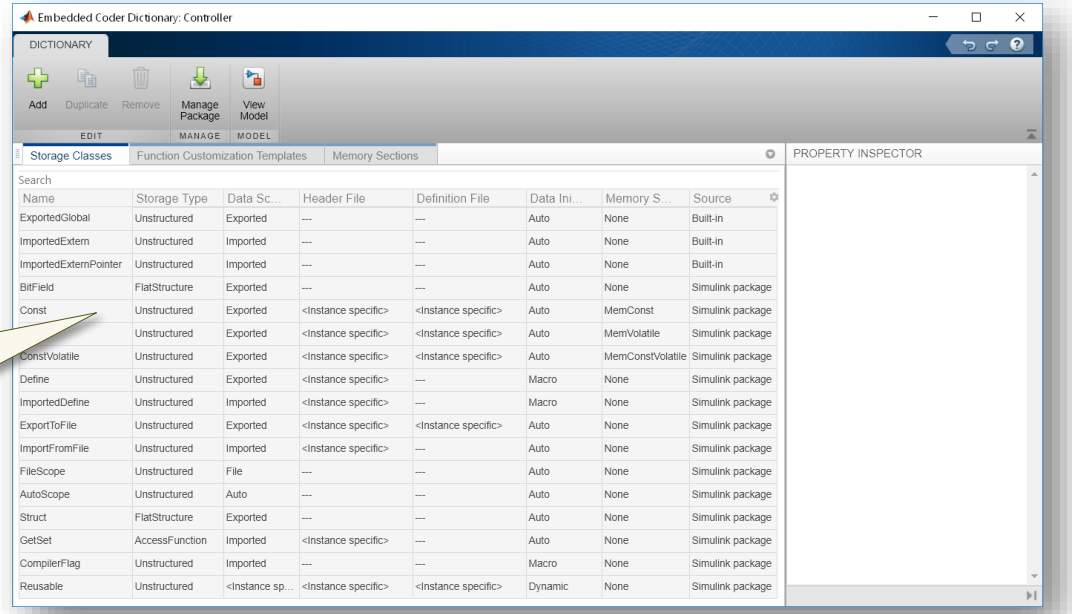


Code Perspective & Embedded Coder Dictionary

- Effective customization for code generation



Code Perspective
- Easy configuration for generated code into any C/C++ SW framework



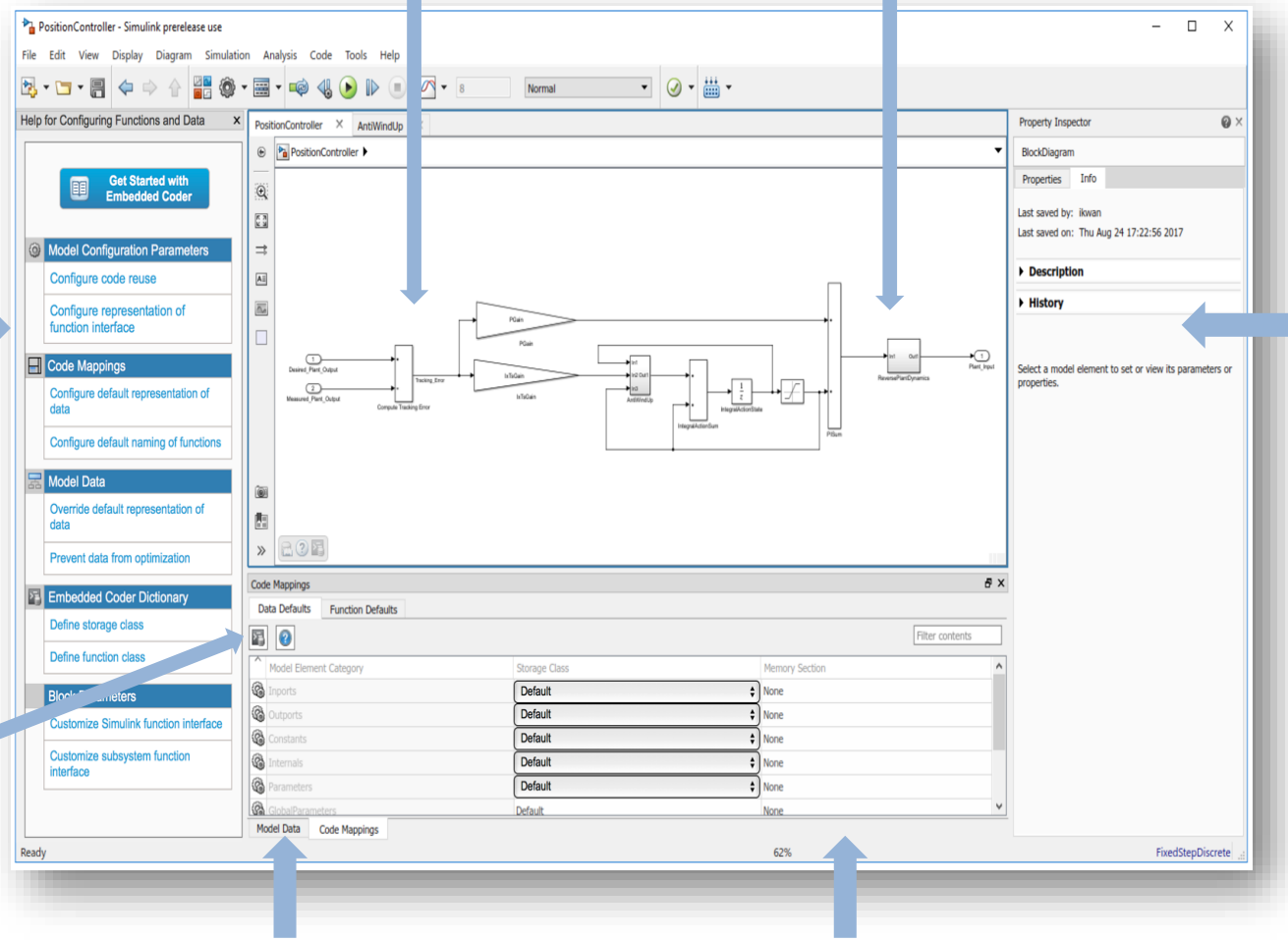
Embedded Coder Dictionary
- GUI for custom code definitions
  - Function template
  - Storage class
  - Memory section

Code Perspective

Edit-time checking

Navigate to code

Help



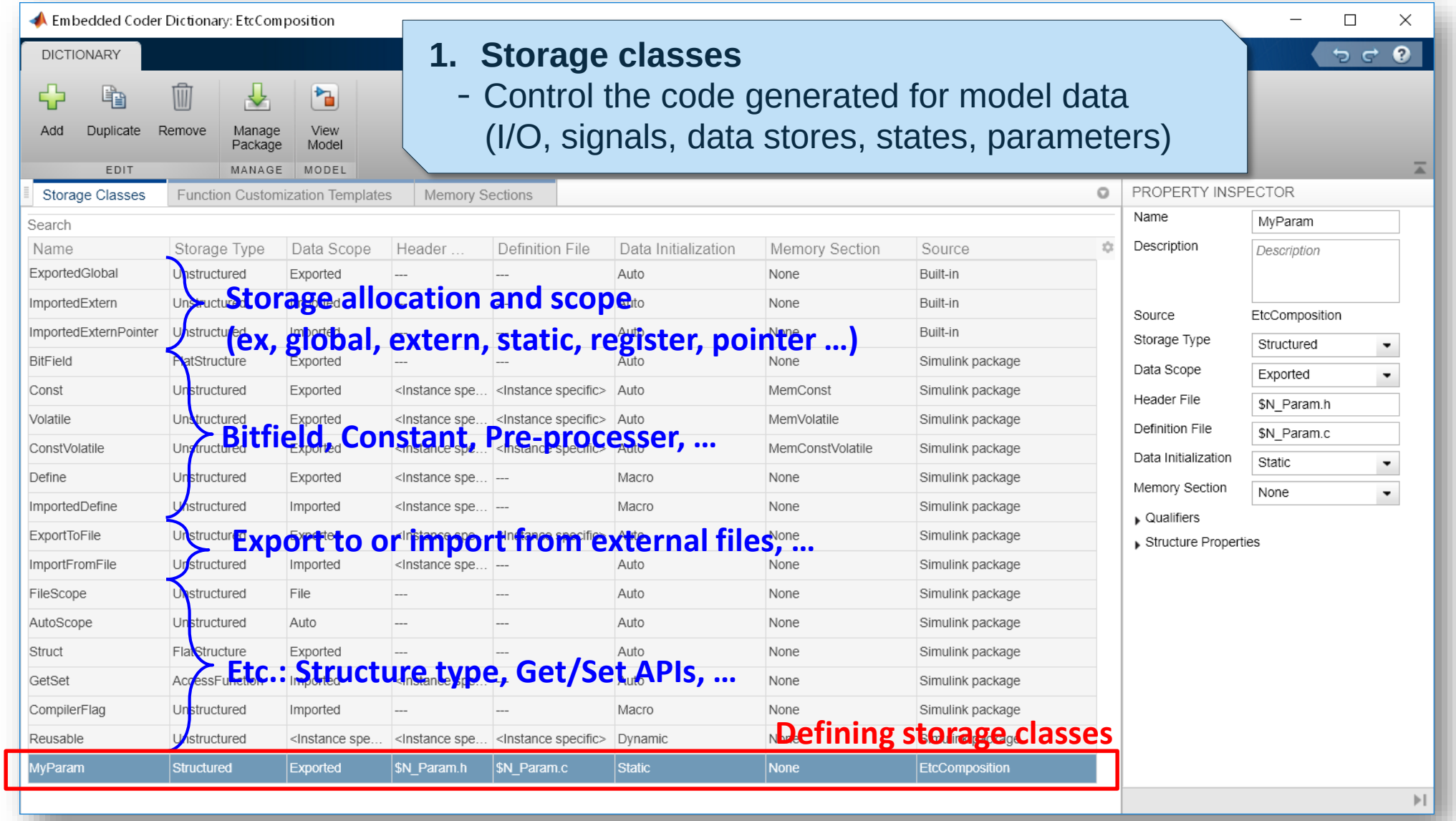
Property Inspector

Code Dictionary

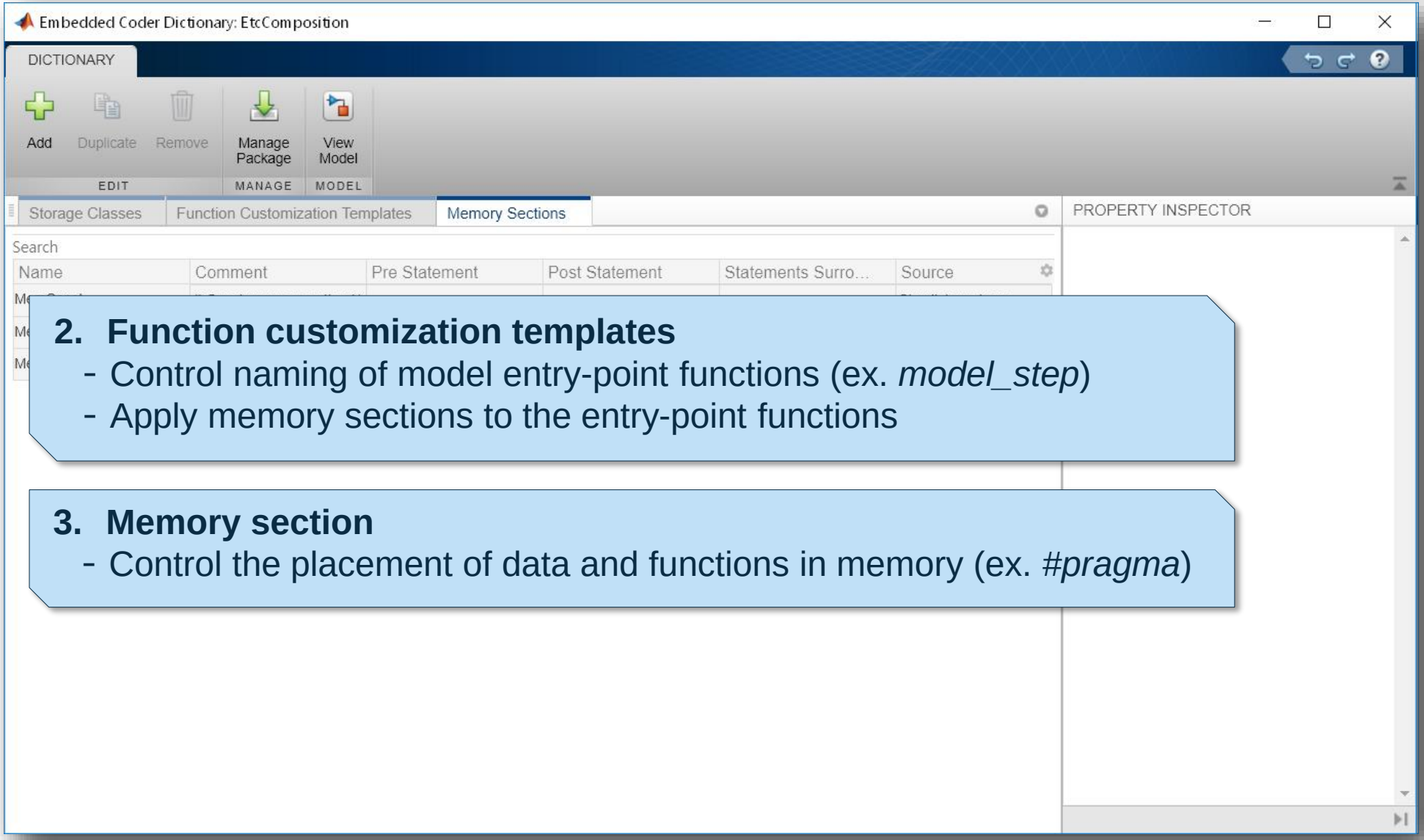
Model Data

Code Mapping

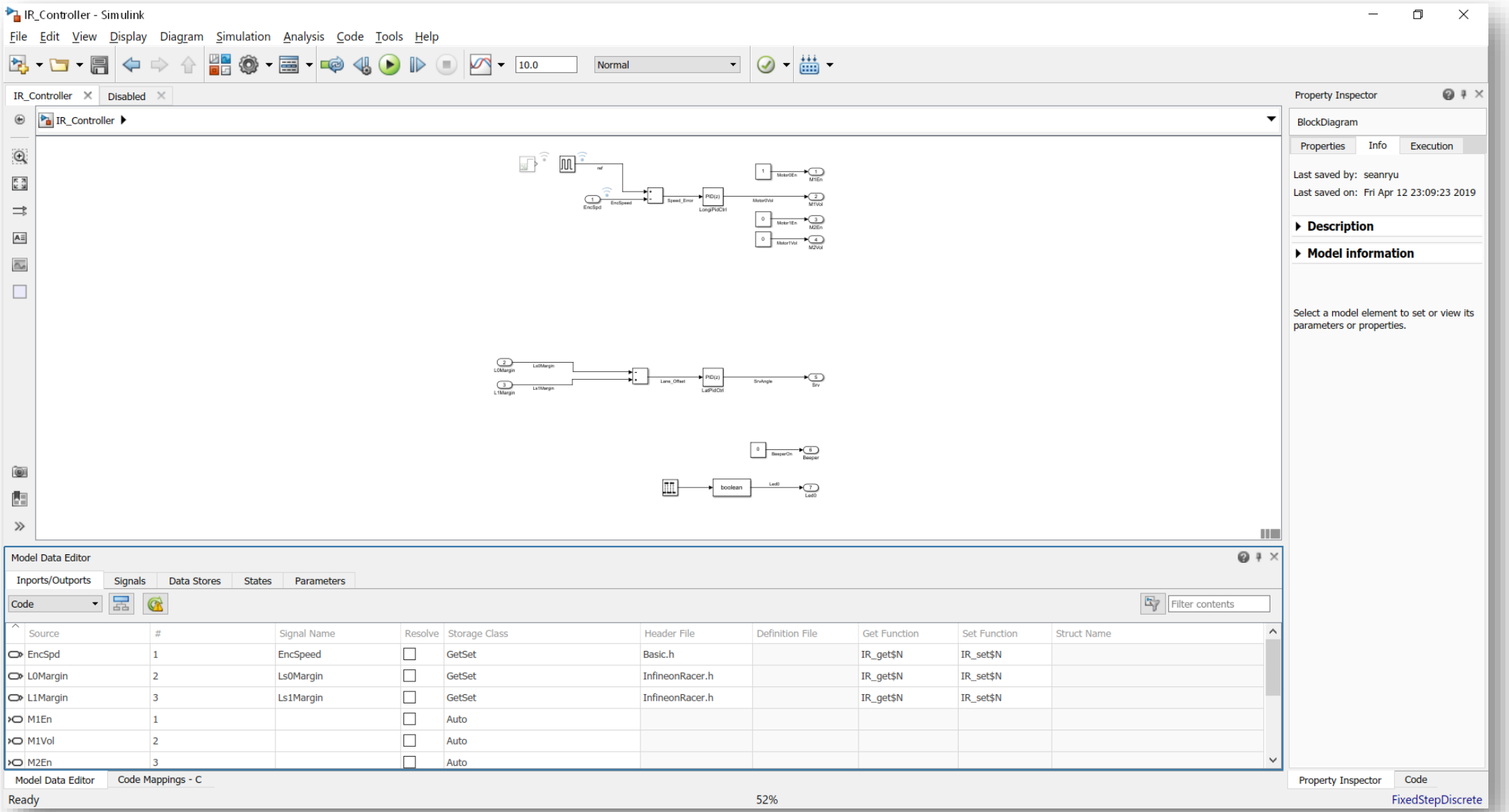
Embedded Coder Dictionary



Embedded Coder Dictionary



Let's Practice…



Automated Driving Testing using Simulink

Challenges

- Algorithm Design
  - Lane Detection
  - Steering Control (Lateral Motion Control)
  - Velocity Control (Longitudinal Motion Control)

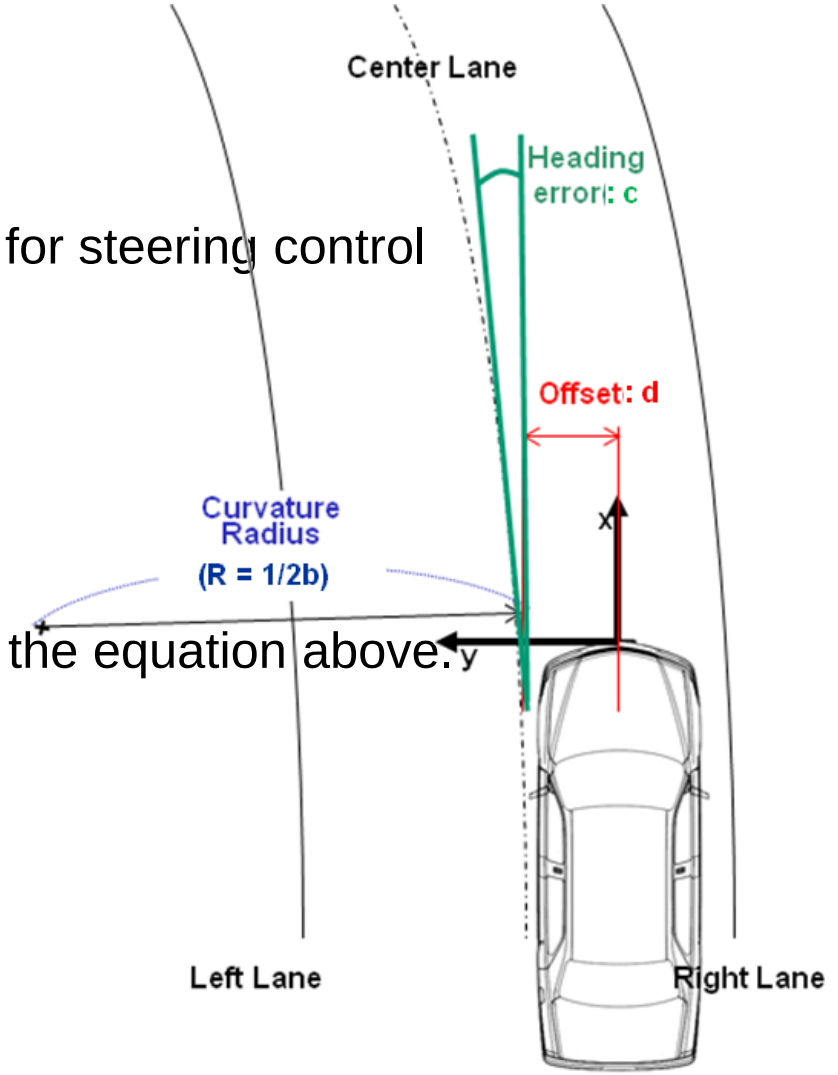
- How can I Test Algorithm in Simulink?
  - Vehicle model
  - Closed Loop Simulation

Lane Detection Processing

Developing Active Safety Systems

Developing the Vision Algorithm Part

- Goal of Lane Detection Algorithm
  - Obtaining the 3rd order polynomial road model for steering control
    - **y = ax^3 + bx^2 + cx + d**
  - But, only information using line scanner is **d** in the equation above.
    - If using two sensor, you can calculate **c**

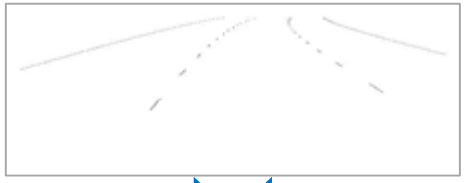


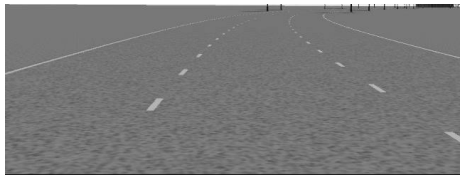
web(fullfile(docroot, 'robotics/ug/pure-pursuit-controller.html'))

Developing Active Safety Systems

Developing the Vision Algorithm Part

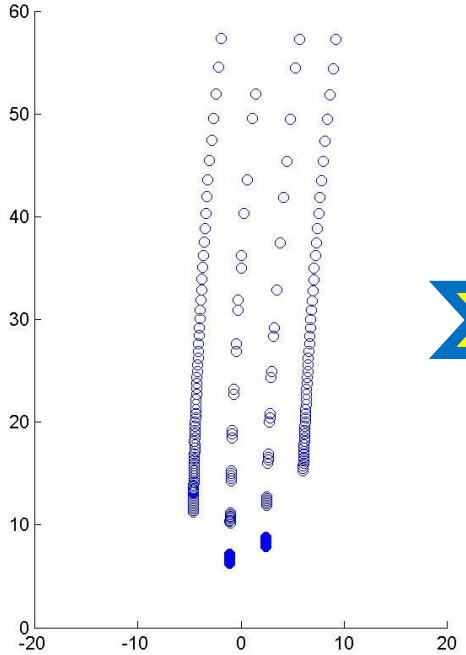
ROI (Region Of Interest)



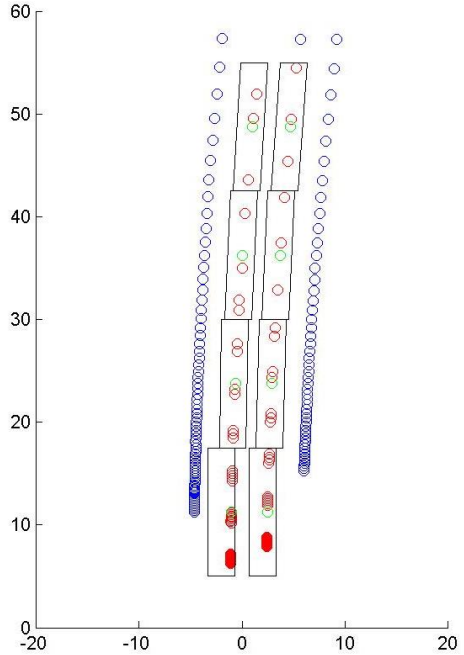


ROI (Region Of Interest)
Cropped Image

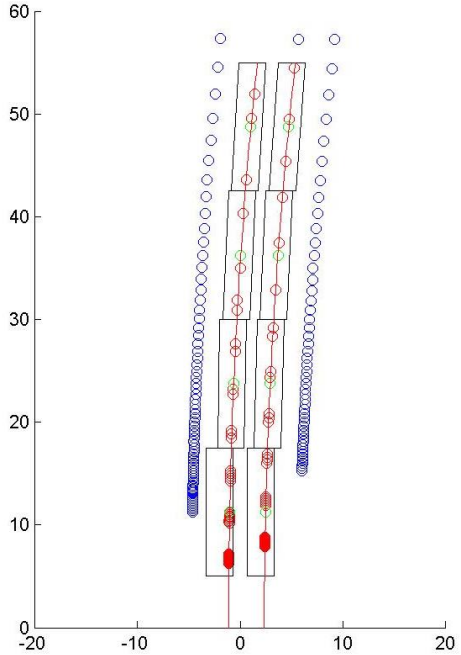
Captured Image



Edge Mapping to World



Selecting Candidate Pts



Curve Fitting

Developing Active Safety Systems

Developing the Vision Algorithm Part

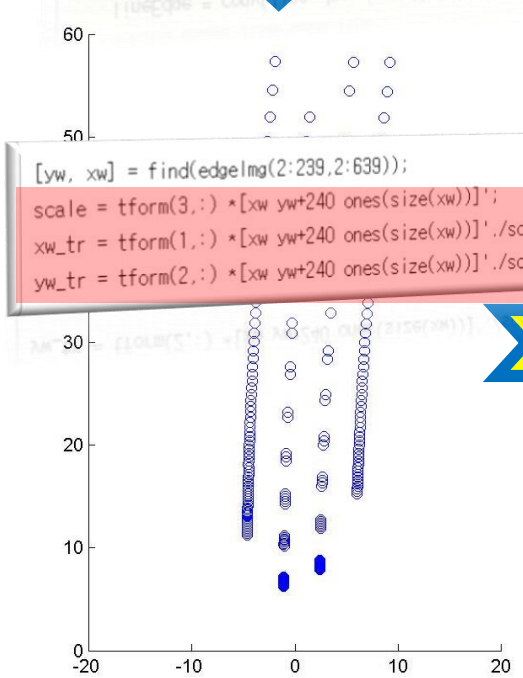
ROI (Region Of Interest)

```
edgeImg = false(size(grayImg));
for lineNum = 1:240
    line = grayImg(lineNum,:);
    kx = filter_kernel(lineNum, 1:floor(ksize));
    ksize = ksize + 0.2;
    % Find edges from each line
    lineEdge = conv(line, kx, 'same') > 0.001;
```

```
%% Detect Edges

grayImg = rgb2gray(im2single(imread('curveImg.bmp')));
grayImg = imcrop(grayImg, [1 241 639 239]);
edgeImg = findLaneEdges(grayImg,kernel);
```

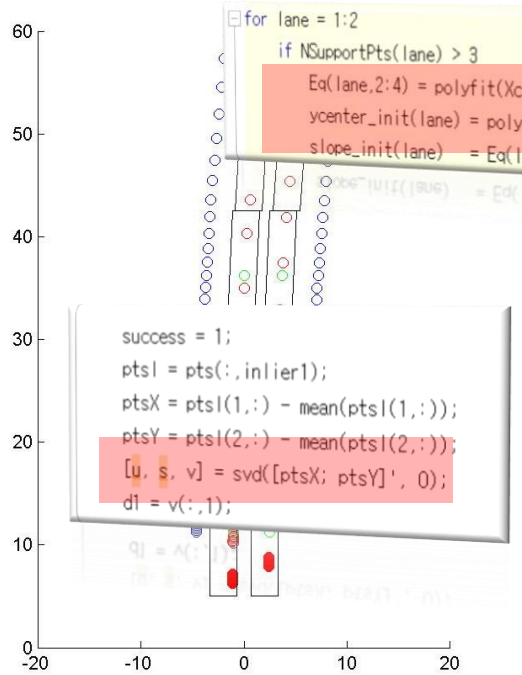
Cropped Image

Captured Image



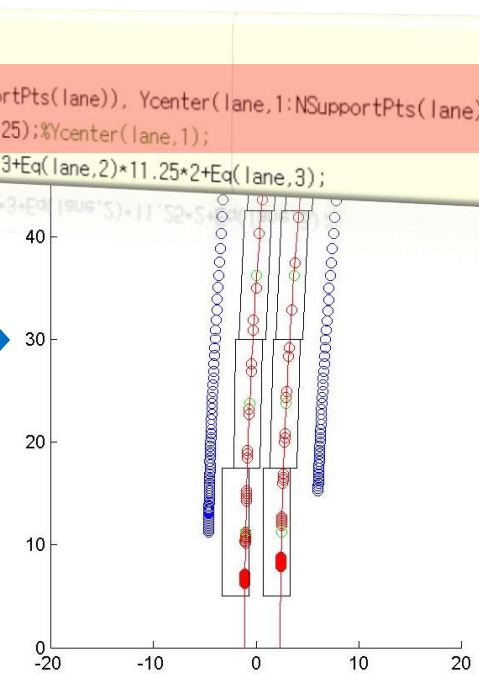
```
[yw, xw] = find(edgeImg(2:239,2:639));
scale = tform(3,:) *[xw yw+240 ones(size(xw))]';
xw_tr = tform(1,:) *[xw yw+240 ones(size(xw))]'./scale;
yw_tr = tform(2,:) *[xw yw+240 ones(size(xw))]'./scale;
```

Edge Mapping to World



```
success = 1;
ptsI = pts(:,inlier1);
ptsX = ptsI(1,:) - mean(ptsI(1,:));
ptsY = ptsI(2,:) - mean(ptsI(2,:));
[u, s, v] = svd([ptsX; ptsY]', 0);
d1 = v(:,1);
```

Selecting Candidate Pts

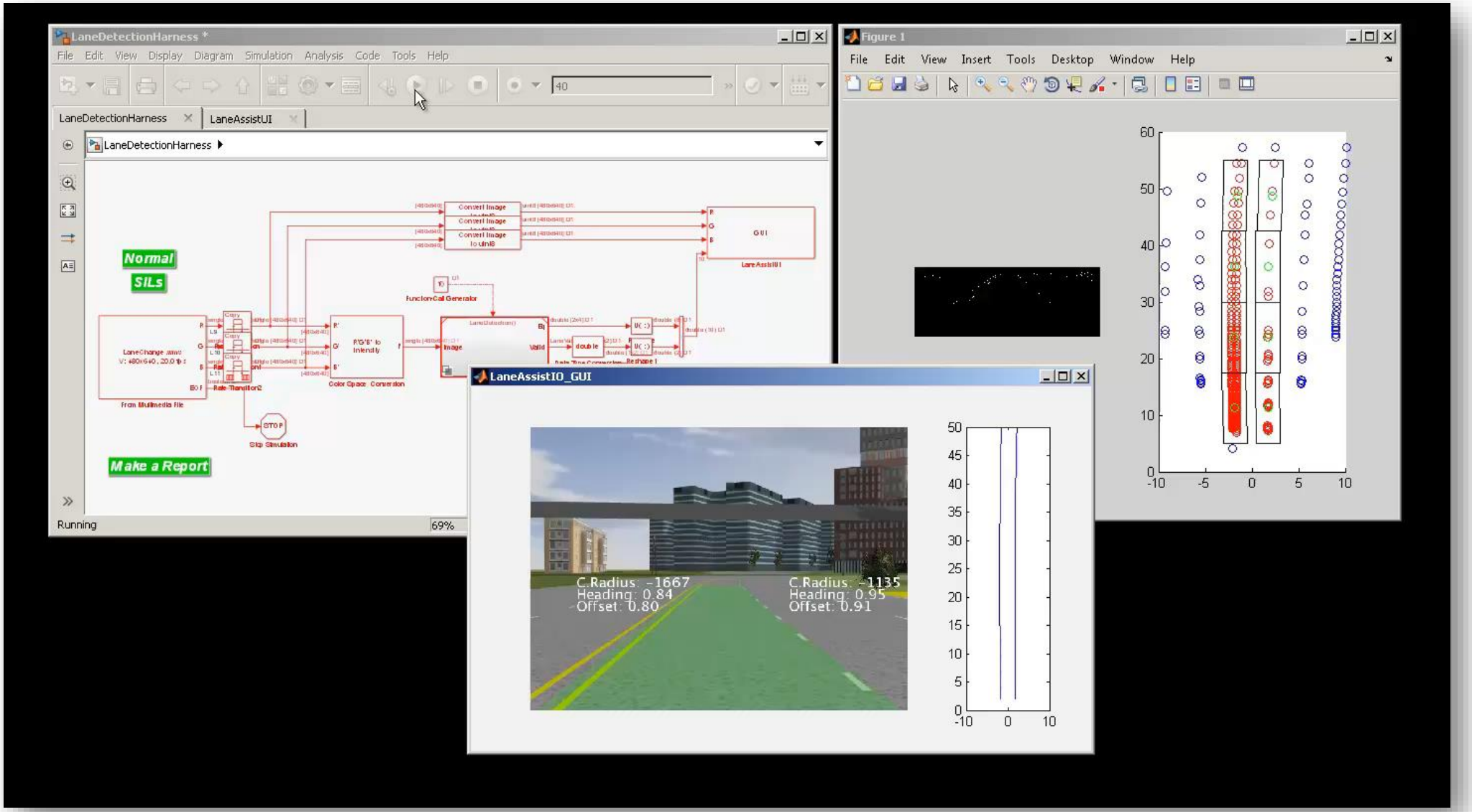


```
for lane = 1:2
    if NSupportPts(lane) > 3
        Eq(lane,2:4) = polyfit(Xcenter(lane,1:NSupportPts(lane)), Ycenter(lane,1:NSupportPts(lane)),2);
        ycenter_init(lane) = polyval(Eq(lane,:), 11.25);%Ycenter(lane,1);
        slope_init(lane)   = Eq(lane,1)*11.25*11.25*3+Eq(lane,2)*11.25*2+Eq(lane,3);
```

Curve Fitting

Developing Active Safety Systems

Developing the Vision Algorithm Part



Computer Vision System Toolbox Examples

MATLAB Examples

Object Detection and Recognition

Automatically Detect and Recognize Text in Natural Images

Detect regions in an image that contain text. This is a common task performed on unstructured scenes. Unstructured scenes are images that

Open Live Script

Detecting Cars Using Gaussian Mixture Models

Detect and count cars in a video sequence using foreground detector based on Gaussian mixture models (GMMs).

Open Live Script

Object Detection in a Cluttered Scene Using Point Feature Matching

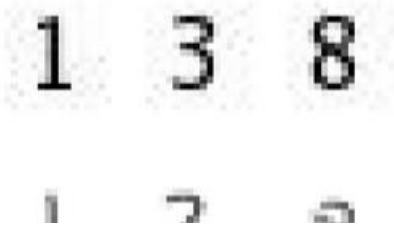
Detect a particular object in a cluttered scene, given a reference image of the object.

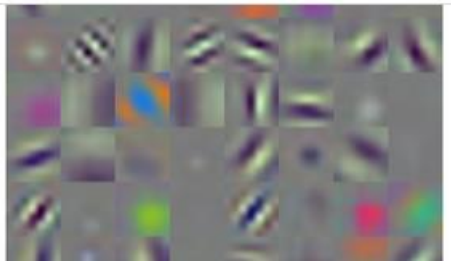
Open Script

Object Detection Using Deep Learning

Train an object detector using deep learning and R-CNN (Regions with Convolutional Neural Networks).

Open Live Script





web(fullfile(docroot, 'vision/index.html'))

Image Processing Toolbox Examples

Deep Learning

Single Image Super-Resolution Using Deep Learning

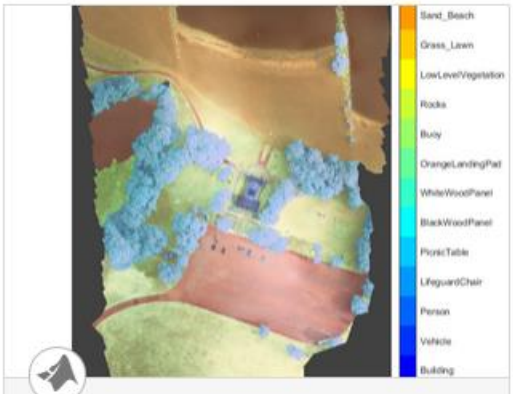
Train a Very-Deep Super-Resolution (VDSR) neural network, then use a VDSR network to estimate a high-resolution image from a single low-

Open Live Script

JPEG Image Deblocking Using Deep Learning

Train a denoising convolutional neural network (DnCNN), then use the network to reduce JPEG compression artifacts in an image.

Open Live Script

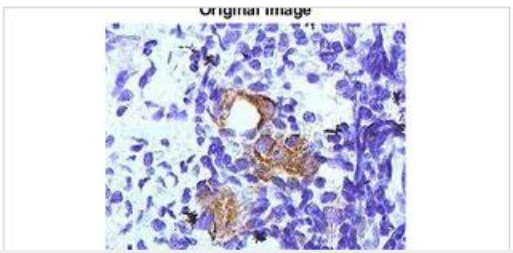


Semantic Segmentation of Multispectral Images Using Deep Learning

Train a U-Net convolutional neural network to perform semantic segmentation of a multispectral image with seven channels: three

Open Live Script

Deblurring

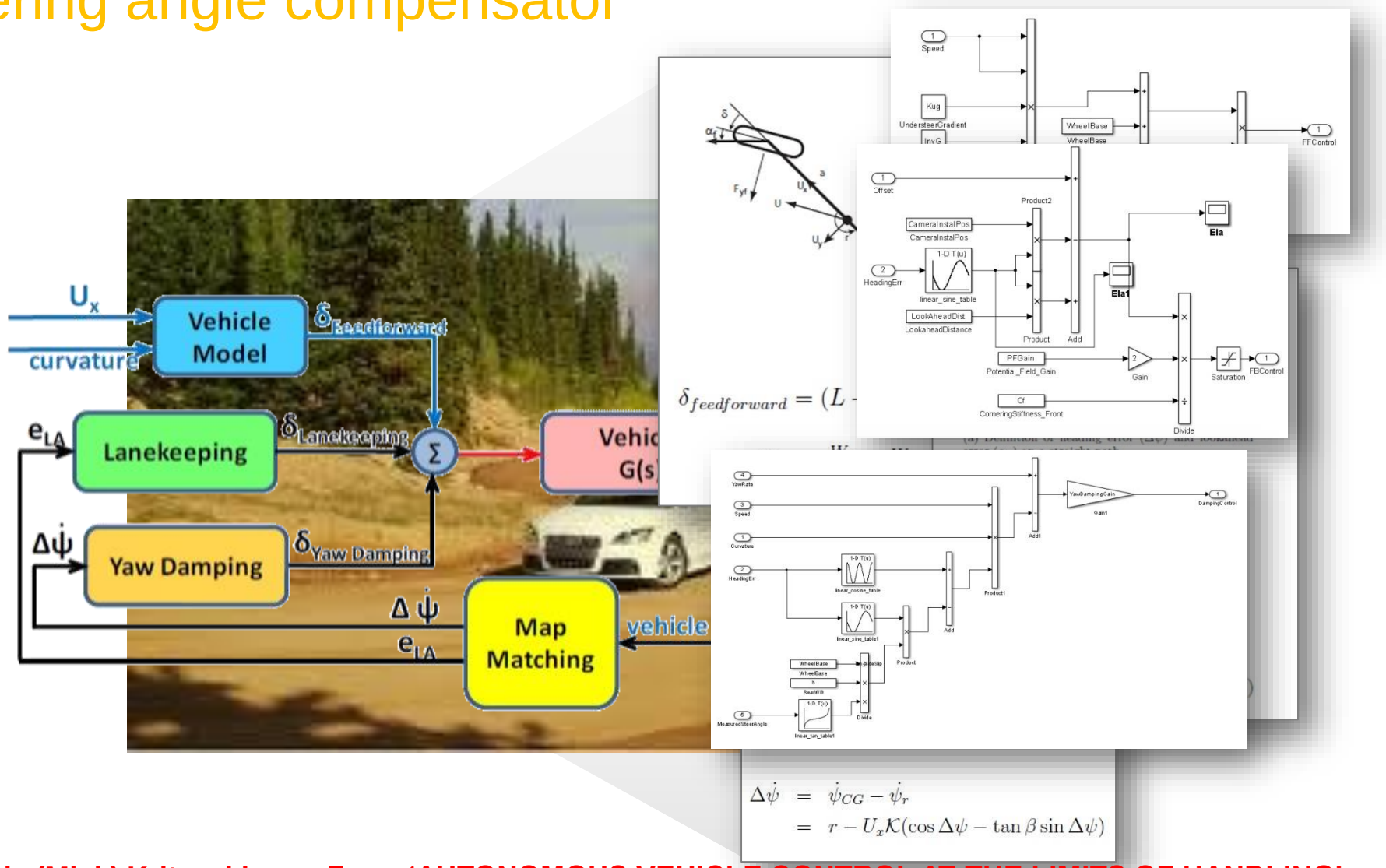


web(fullfile(docroot, 'images/index.html'))

Control Part

Developing Active Safety Systems

Steering angle compensator



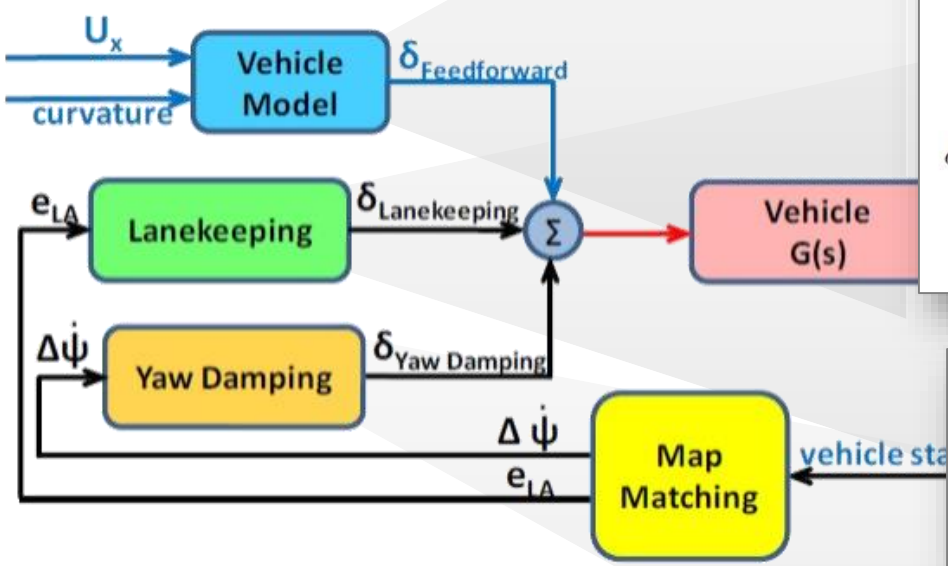
Krisada (Mick) Kritayakirana, From 'AUTONOMOUS VEHICLE CONTROL AT THE LIMITS OF HANDLING', PhD DISSERTATION, Stanford Univ, June 2012

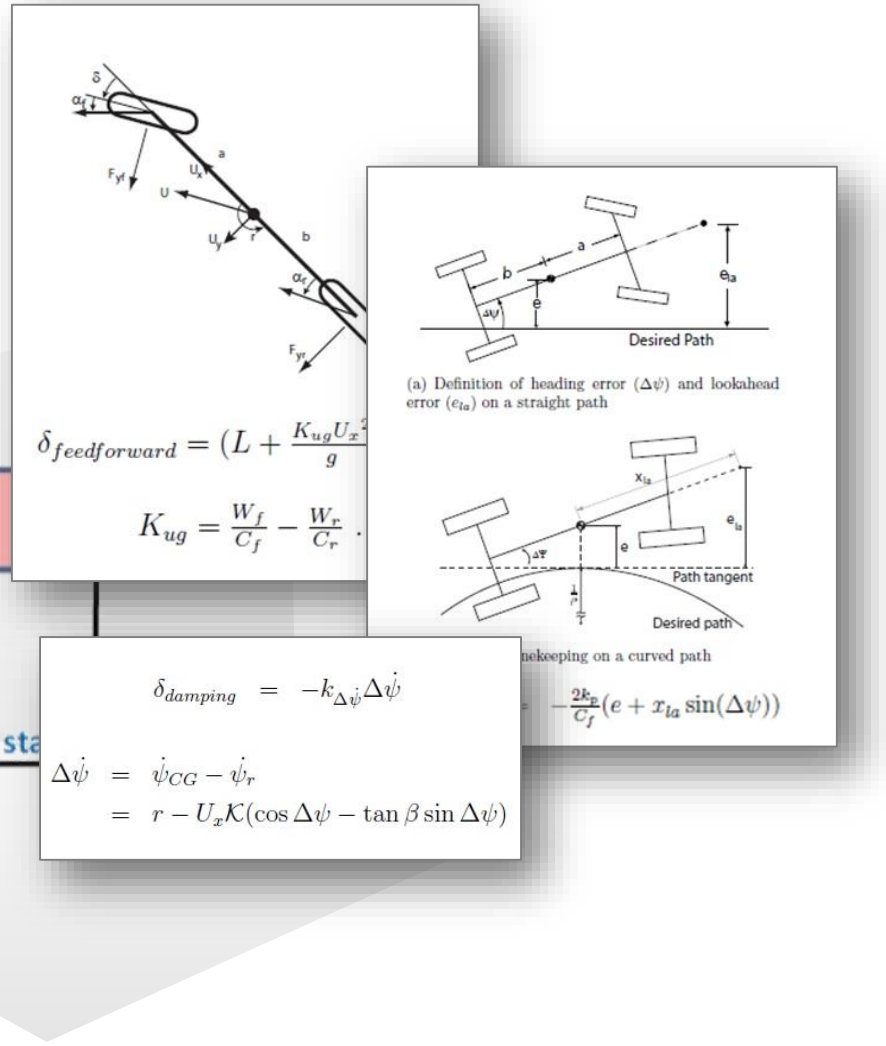
Developing Active Safety Systems

Steering angle compensator

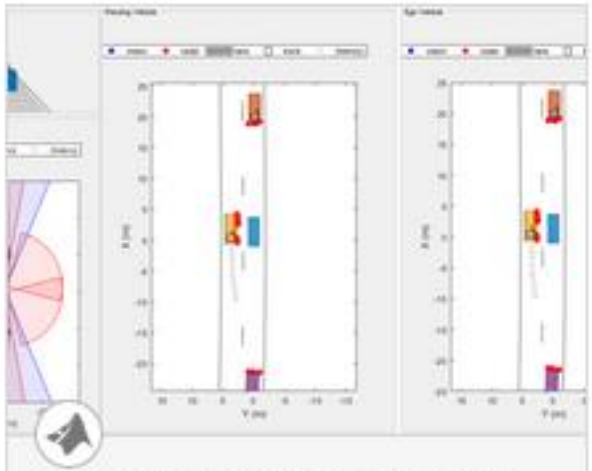
- Curvature information is absent when using line scanner.

➔ Applying feedforward term is impossible.





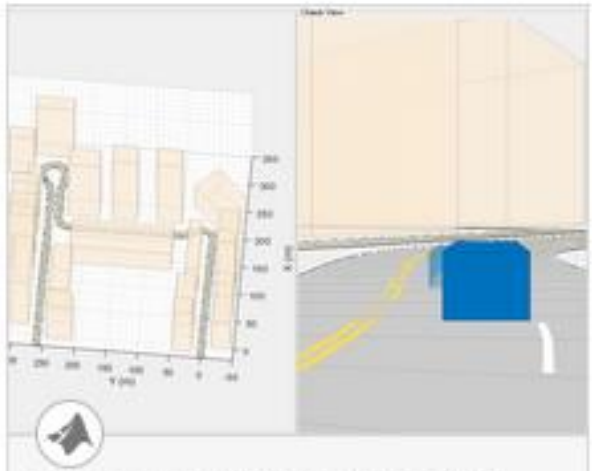
Examples – Sensor Fusion



Extended Object Tracking

Track objects whose dimensions span multiple sensor resolution cells.

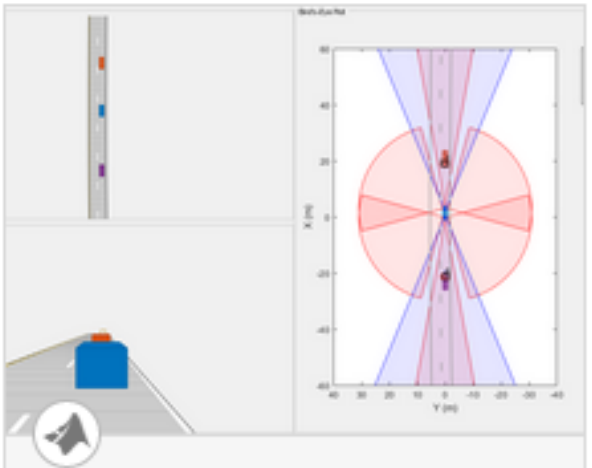
Open Script



Visual-Inertial Odometry Using Synthetic Data

Estimate the pose (position and orientation) of a vehicle by using an inertial measurement unit (IMU) and a monocular camera.

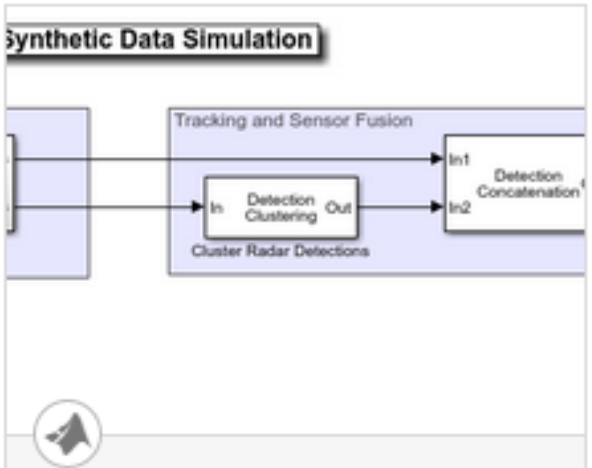
Open Script



Sensor Fusion Using Synthetic Radar and Vision Data

Generate a scenario, simulate sensor detections, and use sensor fusion to track simulated vehicles.

Open Script



Sensor Fusion Using Synthetic Radar and Vision Data in Simulink

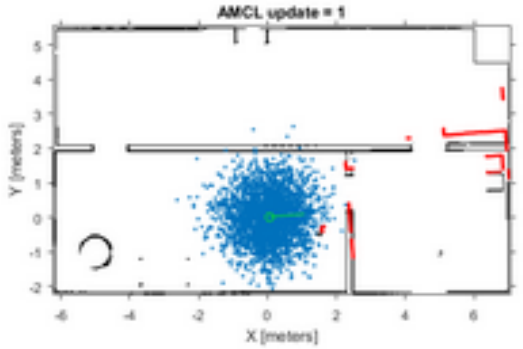
Implement a synthetic data simulation for tracking and sensor fusion in Simulink with Automated Driving System Toolbox.

Open Model

Sensor Fusion and Tracking Toolbox

Automated Driving System Toolbox

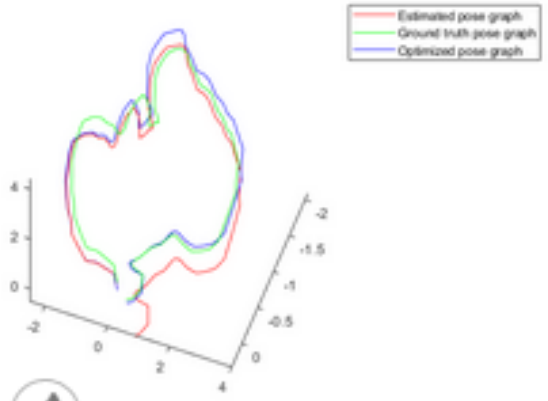
Examples – SLAM and Localization



Localize TurtleBot Using Monte Carlo Localization

Demonstrates an application of the Monte Carlo Localization (MCL) algorithm on TurtleBot® in simulated Gazebo® environment.

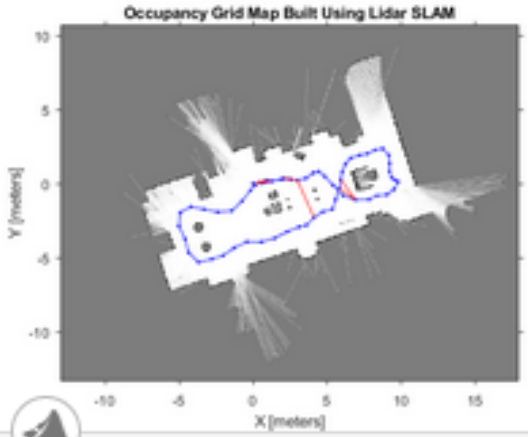
Open Live Script



Reduce Drift in 3-D Visual Odometry Trajectory Using Pose Graphs

Reduce the drift in the estimated trajectory (location and orientation) of a monocular camera using 3-D pose graph optimization.

Open Live Script



Implement Simultaneous Localization And Mapping (SLAM) with Lidar Scans

Demonstrates how to implement the Simultaneous Localization And Mapping (SLAM) algorithm on a collected series of lidar scans using

Open Live Script

Create Occupancy Grid Using Monocular Camera and Semantic...

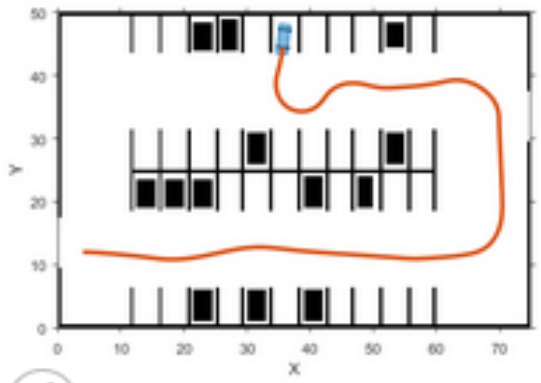
Estimate free space and create an occupancy grid using semantic segmentation and deep learning.

Open Live Script

Robotics System Toolbox

Automated Driving System Toolbox

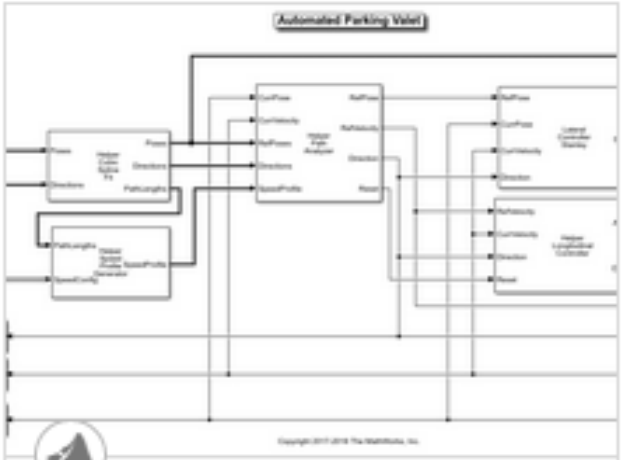
Examples – Path Planning and Following



Automated Parking Valet

Construct an automated parking valet system using path planning, trajectory generation, and vehicle control techniques.

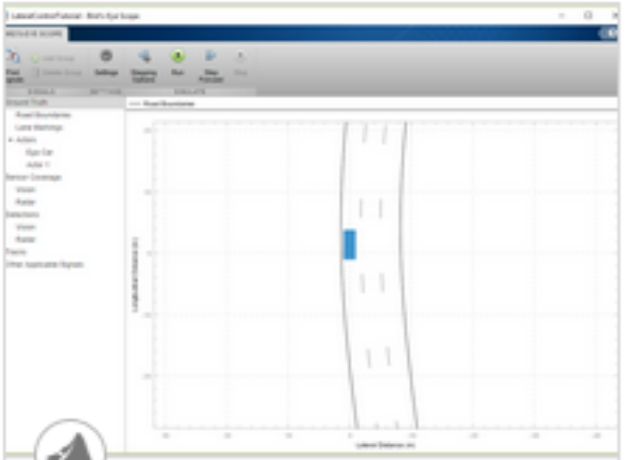
Open Script



Automated Parking Valet in Simulink

Construct an automated parking valet system in Simulink with Automated Driving System Toolbox.

Open Model



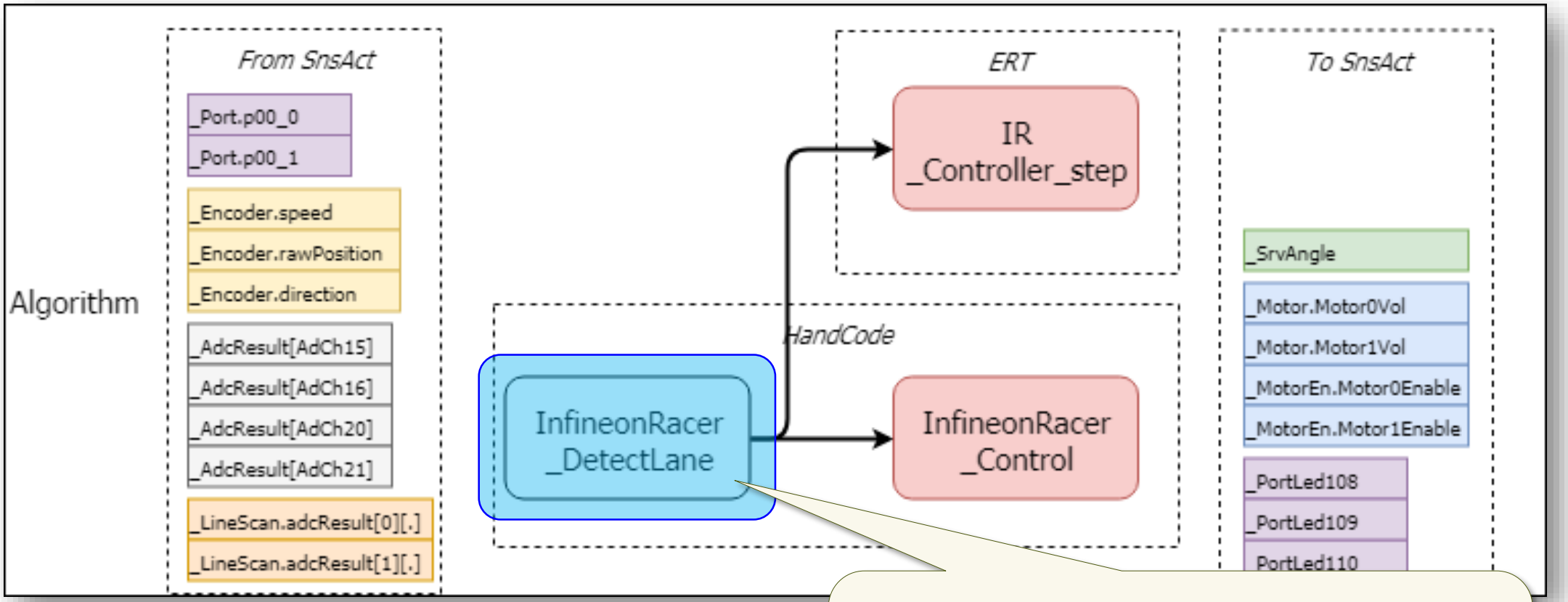
Lateral Control Tutorial

Control the steering angle of a vehicle following a planned path and perform lane changing.

Open Model

Automated Driving System Toolbox

Example for SMCC



Modeling & Code generation for
-. Lane detection
-. Lane keeping
-. Obstacle avoidance

Lane Following Control with Sensor Fusion and Lane Detection

This example shows how to simulate and generate code for an automotive lane-following controller.

In this example, you will:

1. Review a control algorithm that combines sensor fusion, lane detection, and a lane following controller from the Model Predictive Control Toolbox™.
2. Test the control system in a closed-loop Simulink® model using synthetic data generated by the Automated Driving Toolbox™.
3. Configure the code generation settings for software-in-the-loop simulation and automatically generate code for the control algorithm.

This example uses:
Automated Driving Toolbox
Embedded Coder
Model Predictive Control Toolbox
Simulink Control Design
Simulink

Introduction

A lane following system is a control system that keeps the vehicle traveling within a marked lane of a highway, while maintaining a user-set velocity or safe distance from the preceding vehicle. A lane following system includes combined longitudinal and lateral control of the ego vehicle:

- Longitudinal control - Maintain a driver-set velocity and keep a safe distance from the preceding car in the lane by adjus
- Lateral control - Keep the ego vehicle travelling along the centerline of its lane by adjusting the steering of the ego vehic

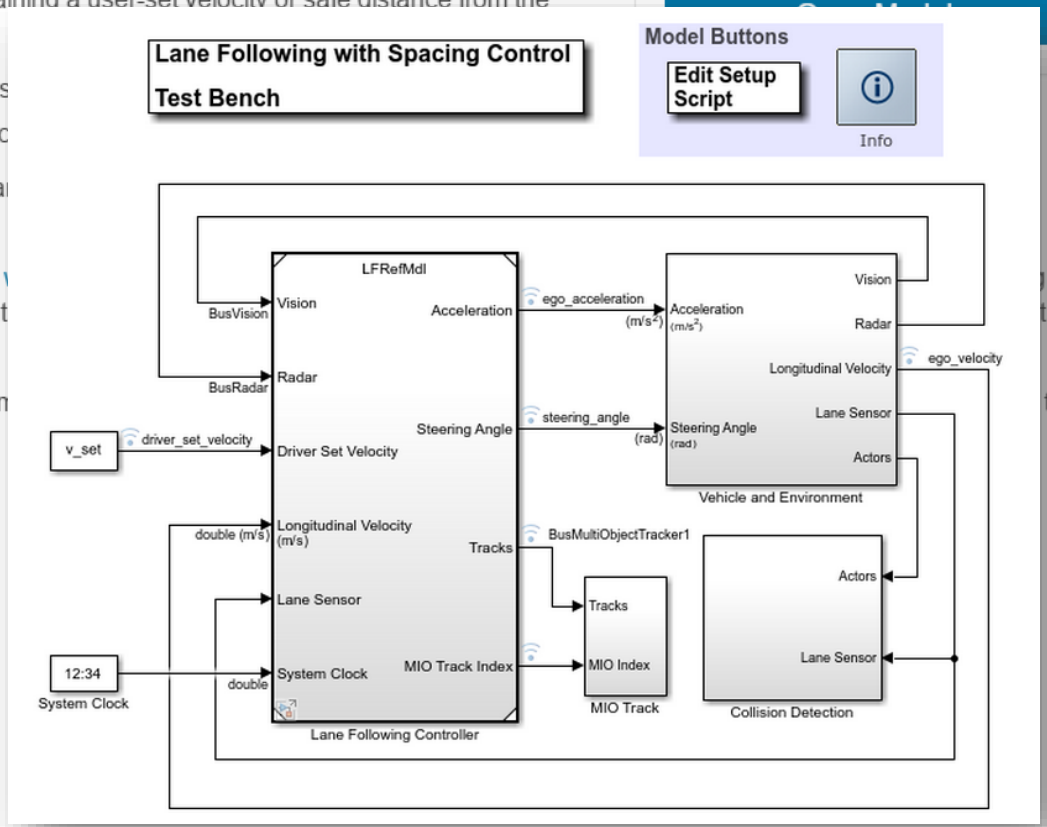
The combined lane following control system achieves the individual goals for longitudinal and lateral control. Further, the la simultaneously.

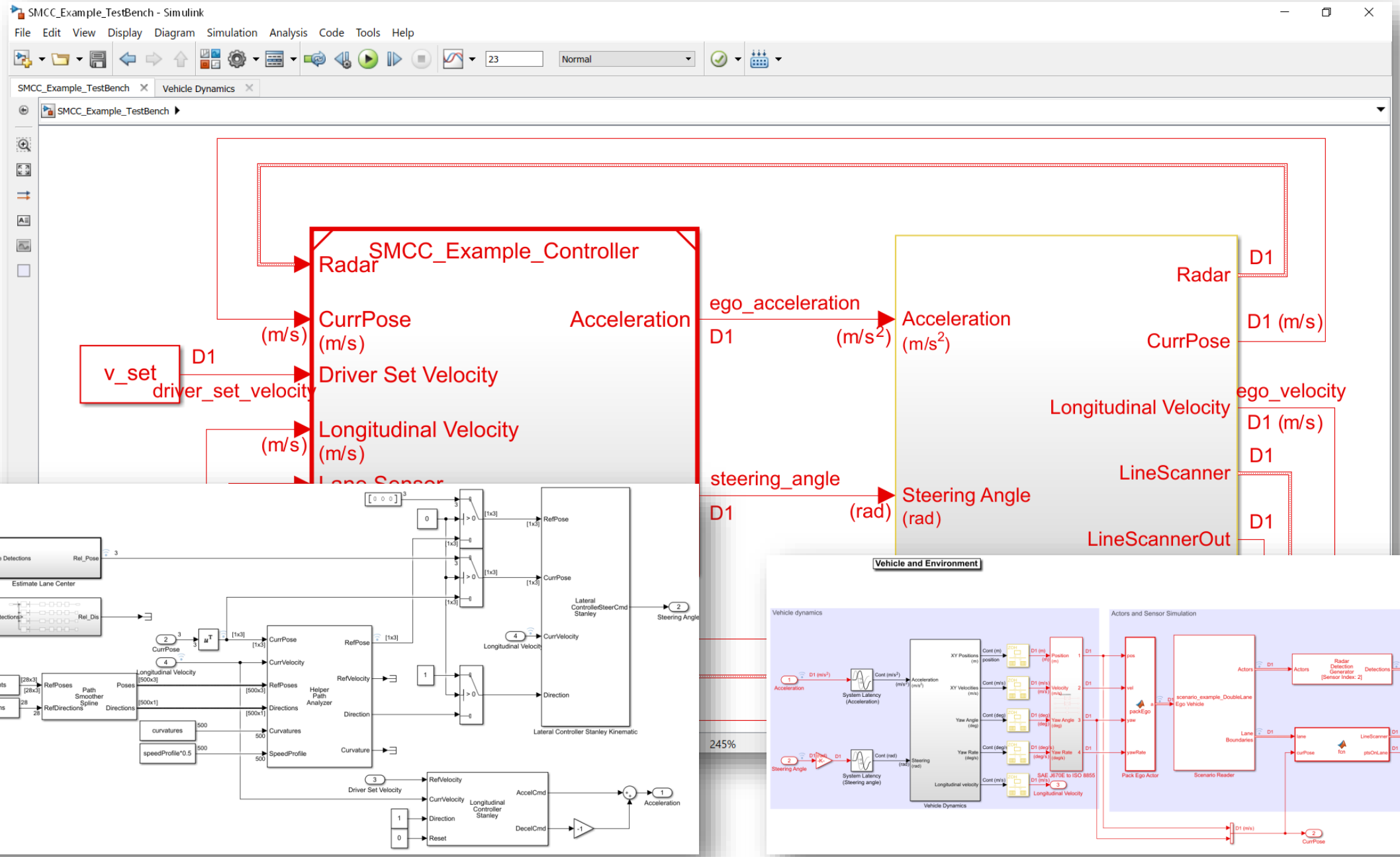
For an example of longitudinal control using adaptive cruise control (ACC) with sensor fusion, see Adaptive Cruise Control lane keeping assist (LKA) system with lane detection, see Lane Keeping Assist with Lane Detection (Model Predictive Cont consider surrounding vehicles.

In this example, both lane detection and surrounding cars are considered. The lane following system synthesizes data fron longitudinal acceleration and steering angle of the ego vehicle.

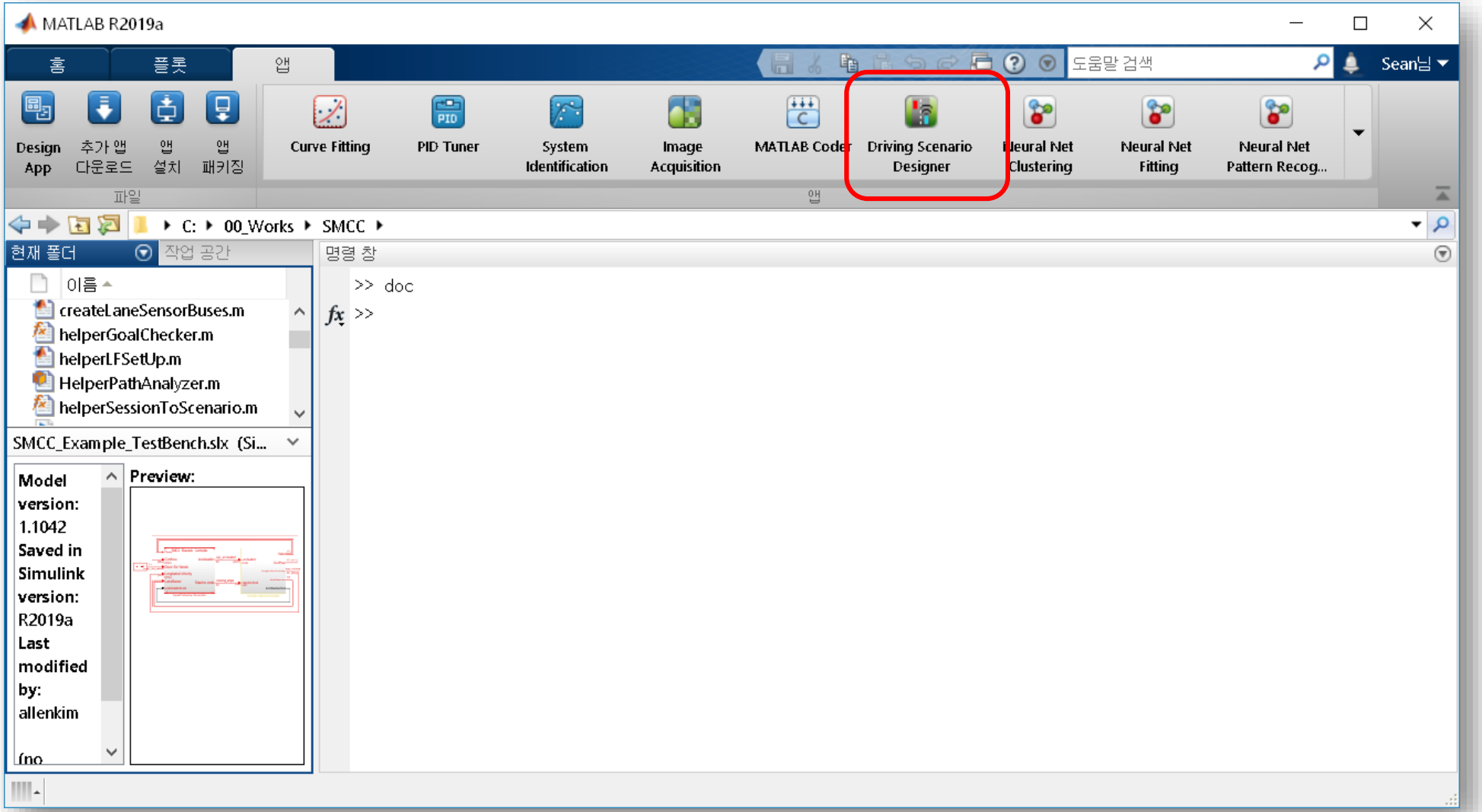
Add example file folder to MATLAB® path.

```
addpath(fullfile(matlabroot,'examples','mpc','main'));
```

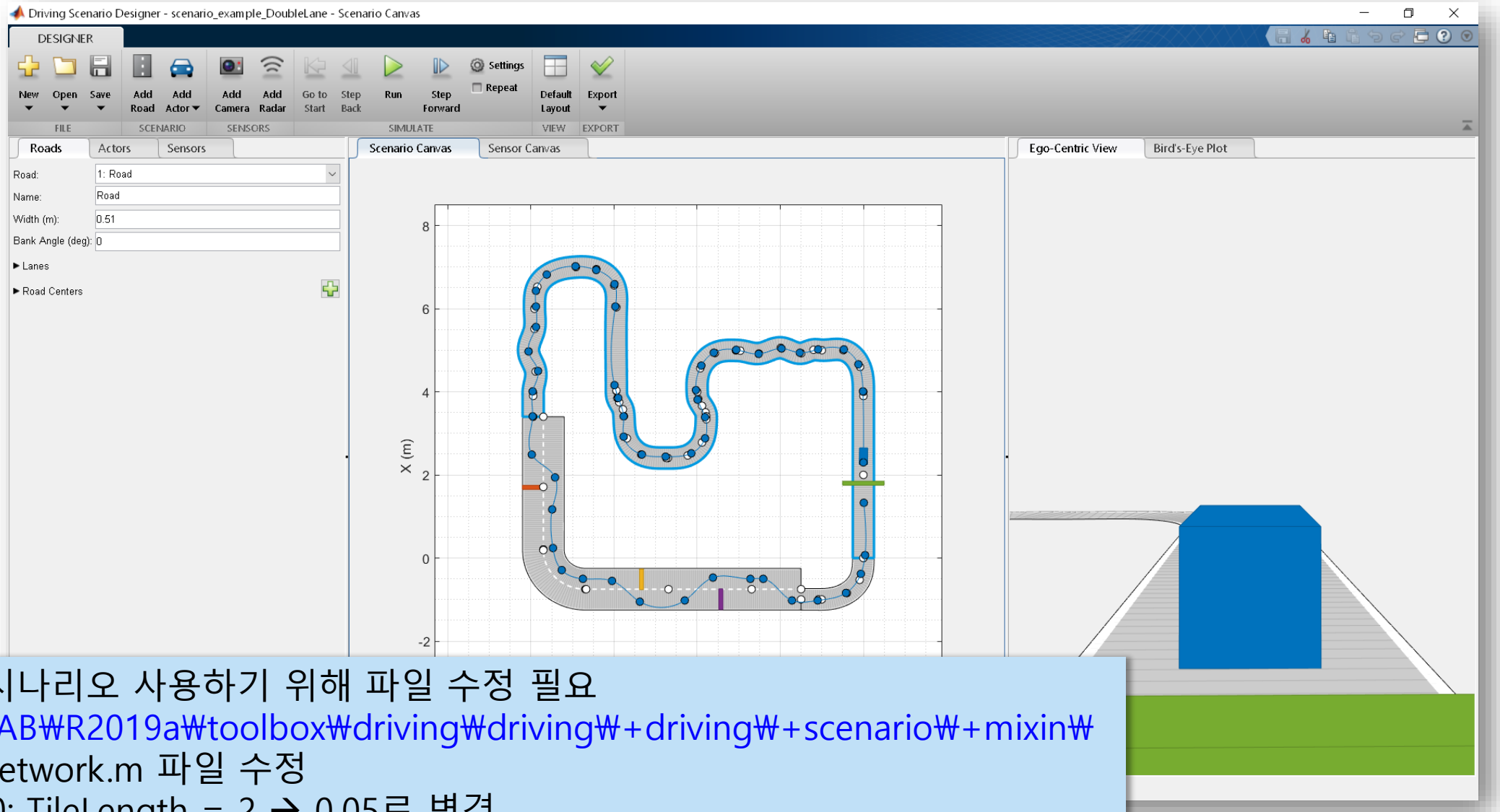



SMCC_Example_TestBench - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
SMCC_Example_TestBench
Vehicle Dynamics
SMCC_Example_Controller
Radar
CurrPose
(m/s)
Driver Set Velocity
Longitudinal Velocity
(m/s)
Acceleration
v_set
D1
driver_set_velocity
ego_acceleration
D1
(m/s²)
Acceleration
(m/s²)
steering_angle
D1
(rad)
Steering Angle
(rad)
Radar
D1
D1 (m/s)
CurrPose
ego_velocity
Longitudinal Velocity
D1 (m/s)
D1
LineScanner
D1
LineScannerOut
Estimate Lane Center
Lane Sensor
LineScannerOut
Lane Detections
Rel_Pose
Rel_Dis
Waypoints
directions
Path Smoother Spline
Helper Path Analyzer
curvatures
speedProfile*0.5
Lateral Controller Stanley Kinematic
Lateral ControllerSteerCmd Stanley
Steering Angle
Longitudinal Velocity
Driver Set Velocity
Longitudinal Controller Stanley
AccelCmd
DecelCmd
Reset
Acceleration
Vehicle and Environment
Vehicle dynamics
Actors and Sensor Simulation
System Latency (Acceleration)
System Latency (Steering angle)
Vehicle Dynamics
Pack Ego Actor
Scenario Reader
Radar Detection Generator [Sensor Index: 2]
245%

Driving Scenario Designer



Driving Scenario Example



*대회용 시나리오 사용하기 위해 파일 수정 필요

C:₩MATLAB₩R2019a₩toolbox₩driving₩driving₩+driving₩+scenario₩+mixin₩

1) RoadNetwork.m 파일 수정

2) Line 20: TileLength = 2 → 0.05로 변경

Control System Design with Simulink®

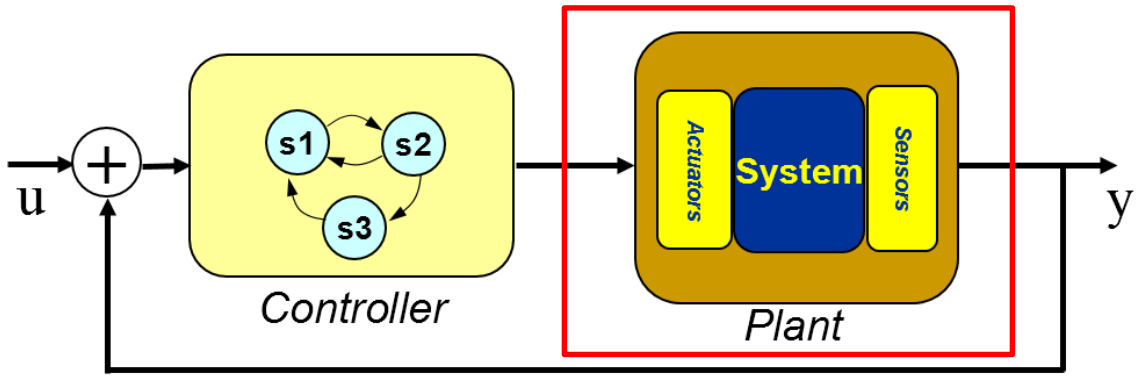
Control Design Process

- **Modeling:**

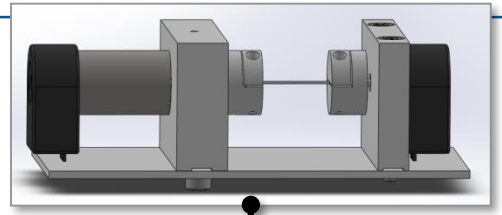
  understanding your design problem

- **Control Design:**

  solving your design problem



Modeling Approaches



First Principles

Data-Driven

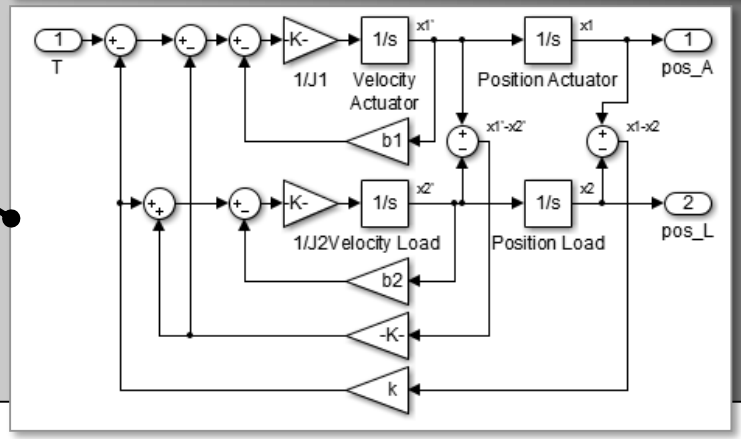
Programming

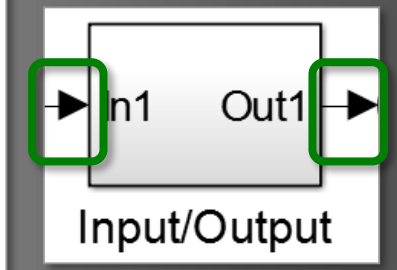
Block Diagram

Modeling Language

Symbolic Methods

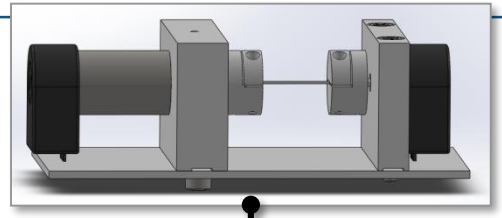
$$J_1 x_1'' = -b_1 x_1' - k(x_1 - x_2) - b_{12}(x_1' - x_2') + T$$
$$J_2 x_2'' = -b_2 x_2' + k(x_1 - x_2) - b_{12}(x_1' - x_2')$$

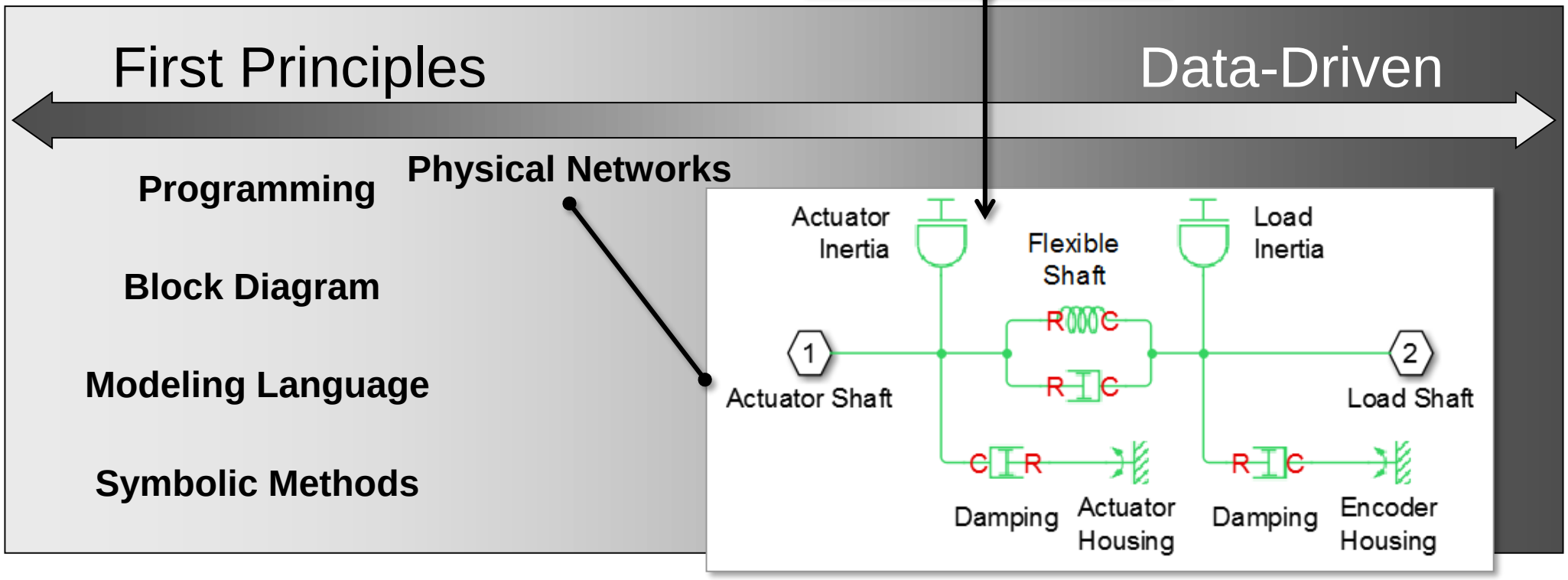




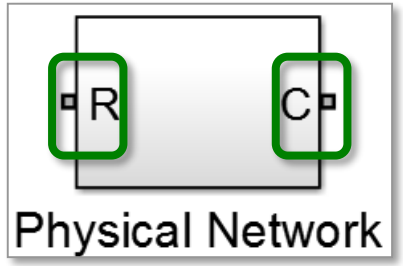
- Purpose: Explore design or physical parameters
- Requirements:
  - Physics of system are well-known
  - System-level equations can be derived and implemented

Modeling Approaches

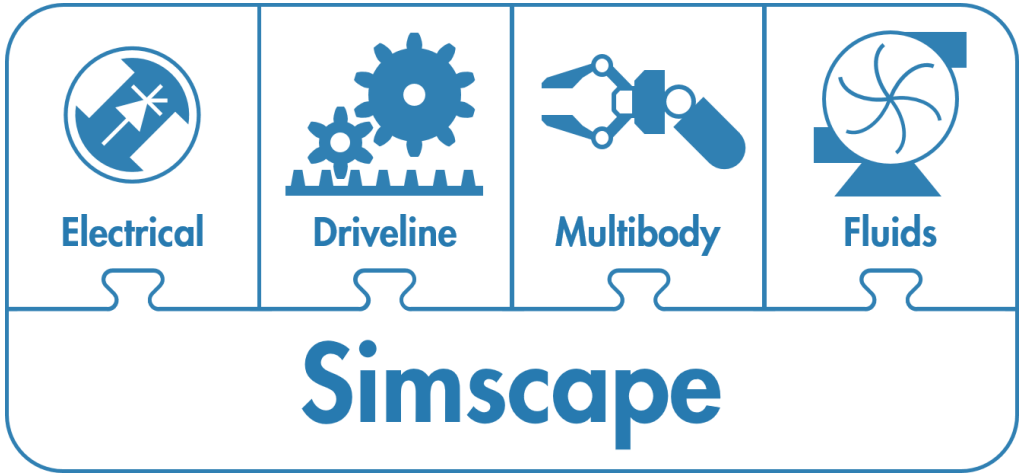




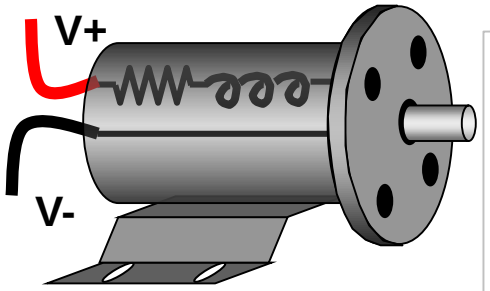
- Purpose: Explore design or physical parameters
- Requirements:
  - Physics of system are well-known
  - Component-level models exist or can be created

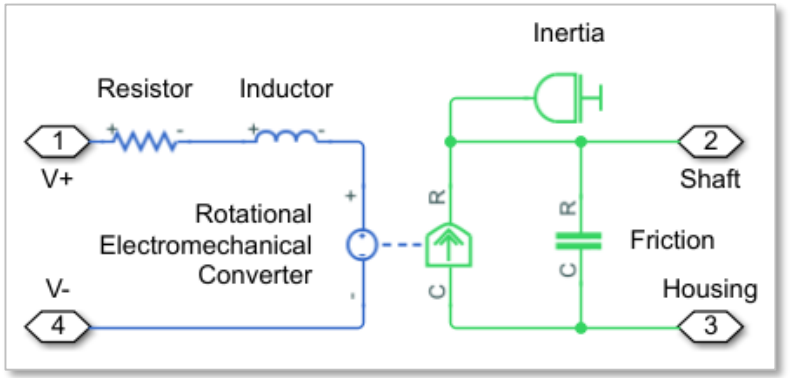


Simscape
Overview



- Enables physical modeling (acausal) of multidomain physical systems
  - Assemble a schematic
  - Equations derived automatically
  - Leverage MATLAB and Simulink

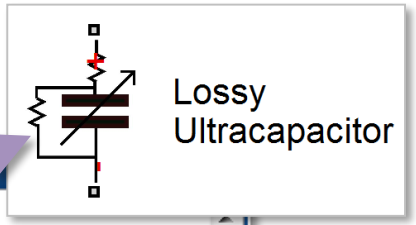




- With Simscape you can:
  - Refine requirements for system
  - Discover integration issues early
  - Design control systems and logic
  - Optimize system-level performance
  - Test embedded software without hardware prototypes

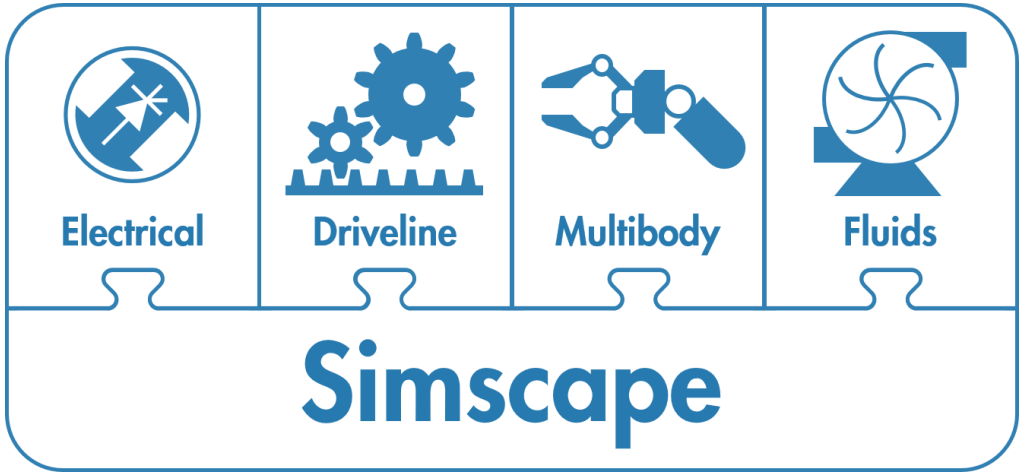
$$i = (C_0 + C_v v)\frac{dv}{dt} + \frac{v}{r_d}$$

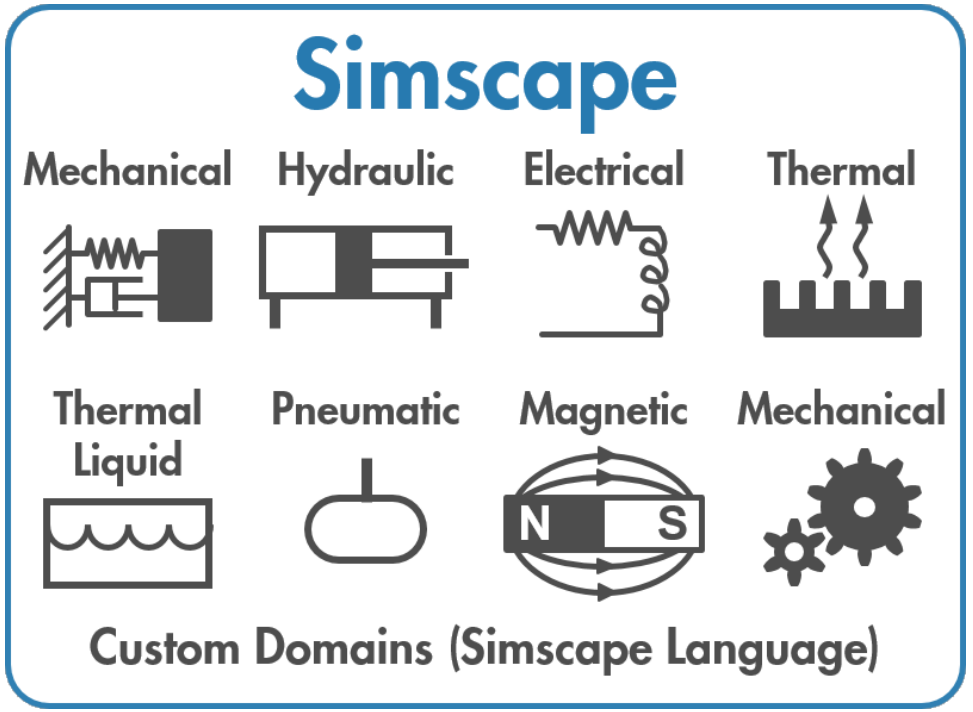
```
Editor - C:\+MyComponent...ossyUltraCapacitor.ssc
40  equations
41    i == (C0 + Cv*vc)*vc.der + vc/Rd;
42    v == vc + i*R;
43  end
```



Simscape Products

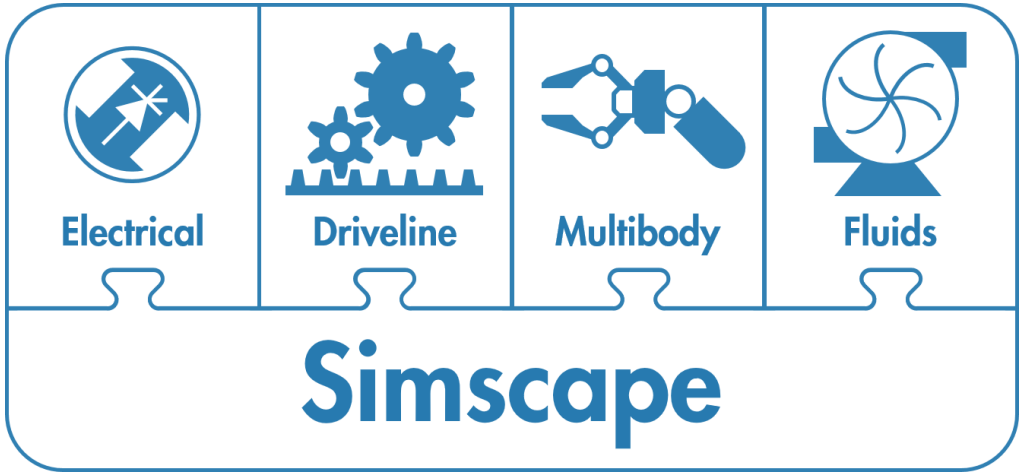
- Simscape platform
  - Foundation libraries in 8 domains
  - Language for defining custom blocks
    - Extension of MATLAB
  - Simulation engine and custom diagnostics
- Simscape add-on libraries
  - Extend foundation domains with components, effects, parameterizations
  - Models can be converted to C code

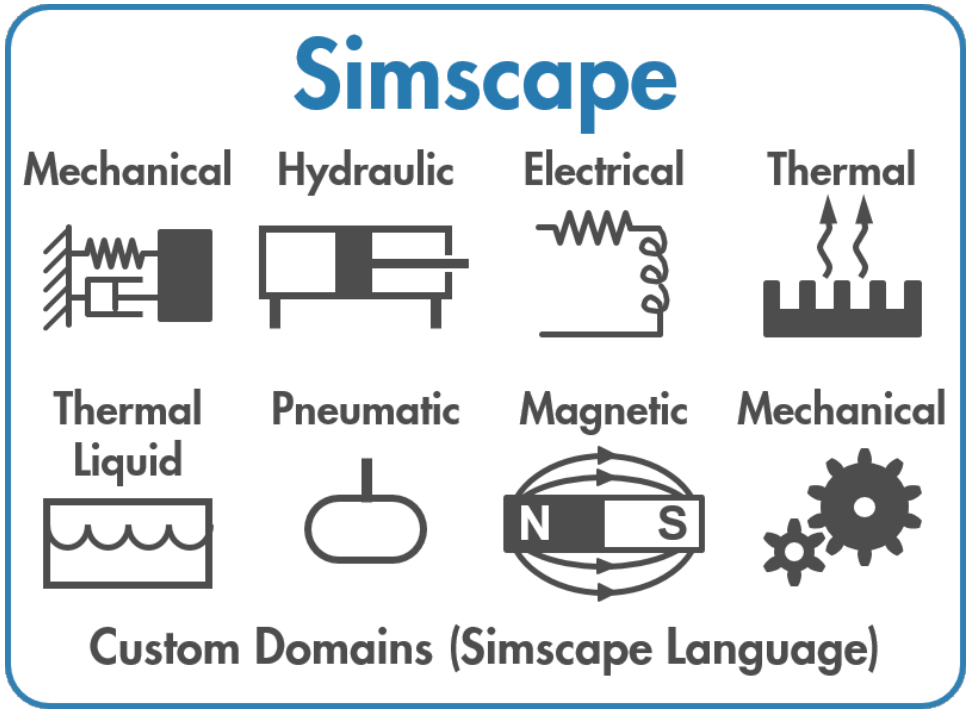




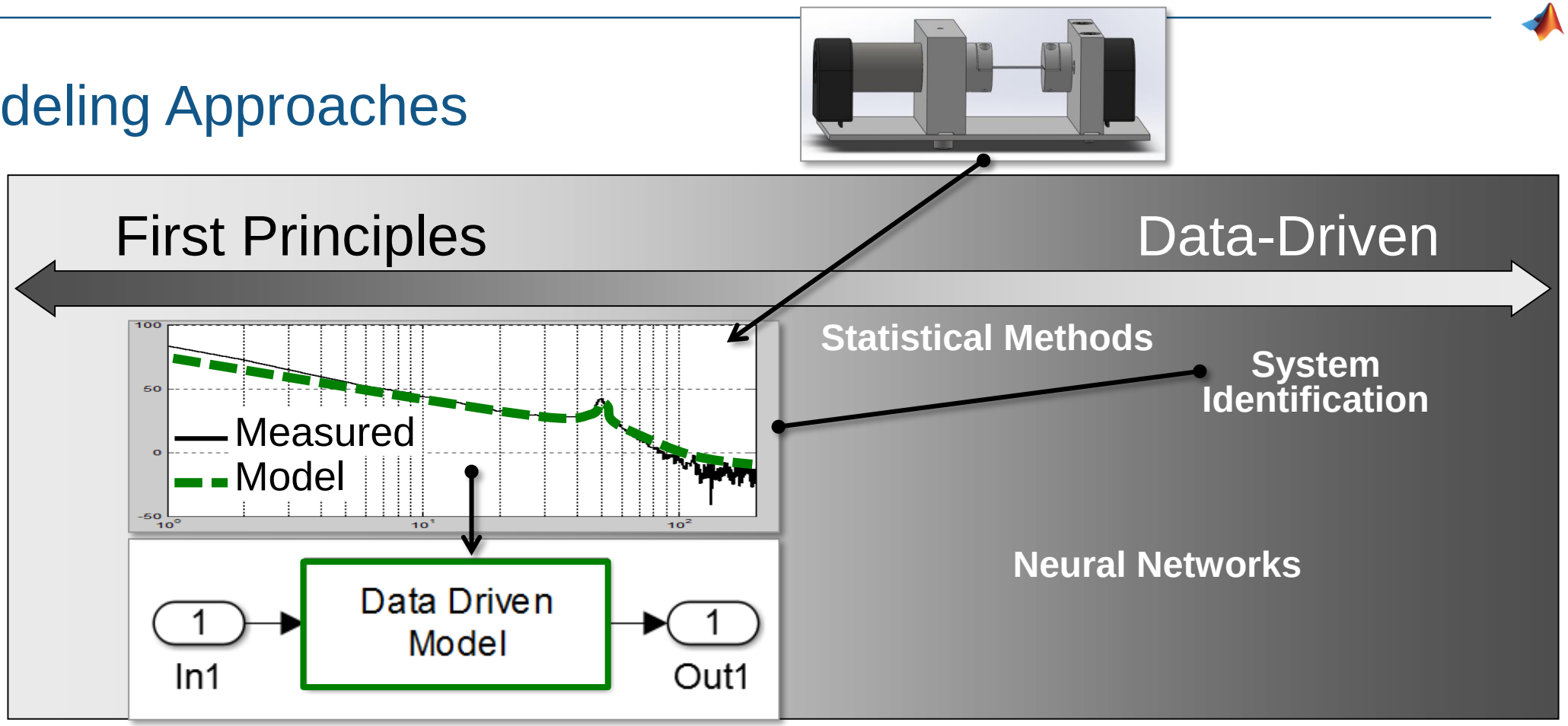
Simscape Add-on Libraries

- Simscape Electricals
  - Actuators, sensors, and semiconductors
  - Three-phase electrical networks
- Simscape Fluids
  - Pumps, actuators, pipelines, valves, tanks
- Simscape Multibody
  - Multibody systems: joints, bodies, frames
- Simscape Driveline
  - Gears, leadscrew, clutches, tires, engines





Modeling Approaches



- Purpose: Model an existing design (real or virtual)
- Requirements:
  - Relevant set of measured data is available
  - Design and physical parameters will not be changed

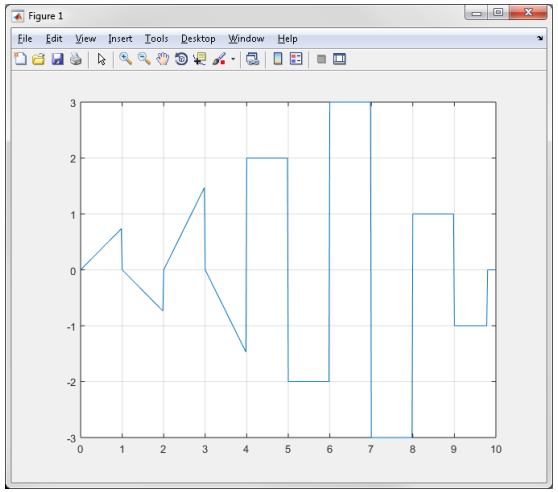
web(fullfile(docroot, 'ident/index.html'))

Modeling Approach for unknown Actuator

Data-Driven Approach : System Identification

Input

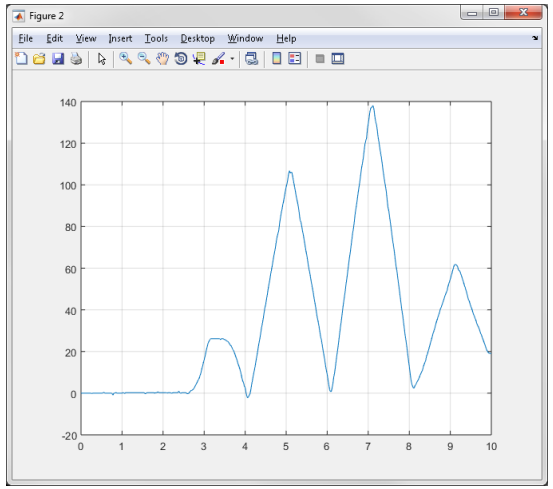
Voltage



System

Output

Angle or
Angular Velocity



>>load measuredData.mat

Modeling Approach for unknown Actuator

Data-Driven Approach : System Identification

u

Input

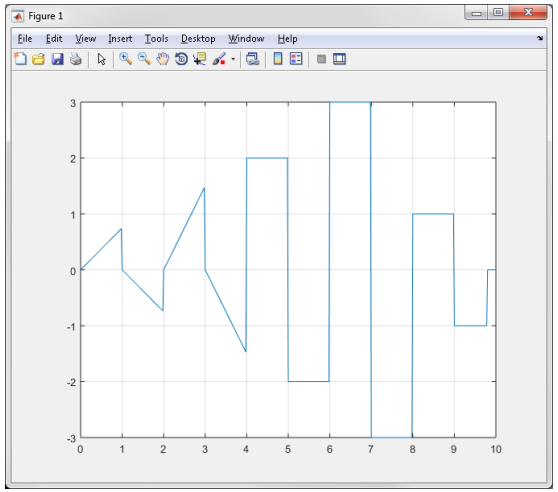
Voltage

$$x(t+1) = Fx(t) + Gu(t)$$
$$y = Hx(t)$$

y

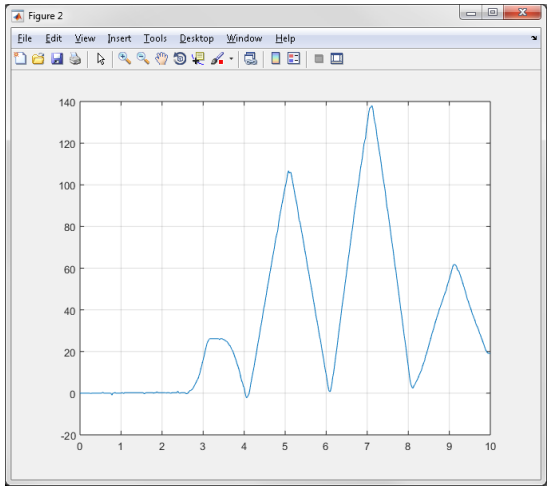
Output

Angle or
Angular Velocity



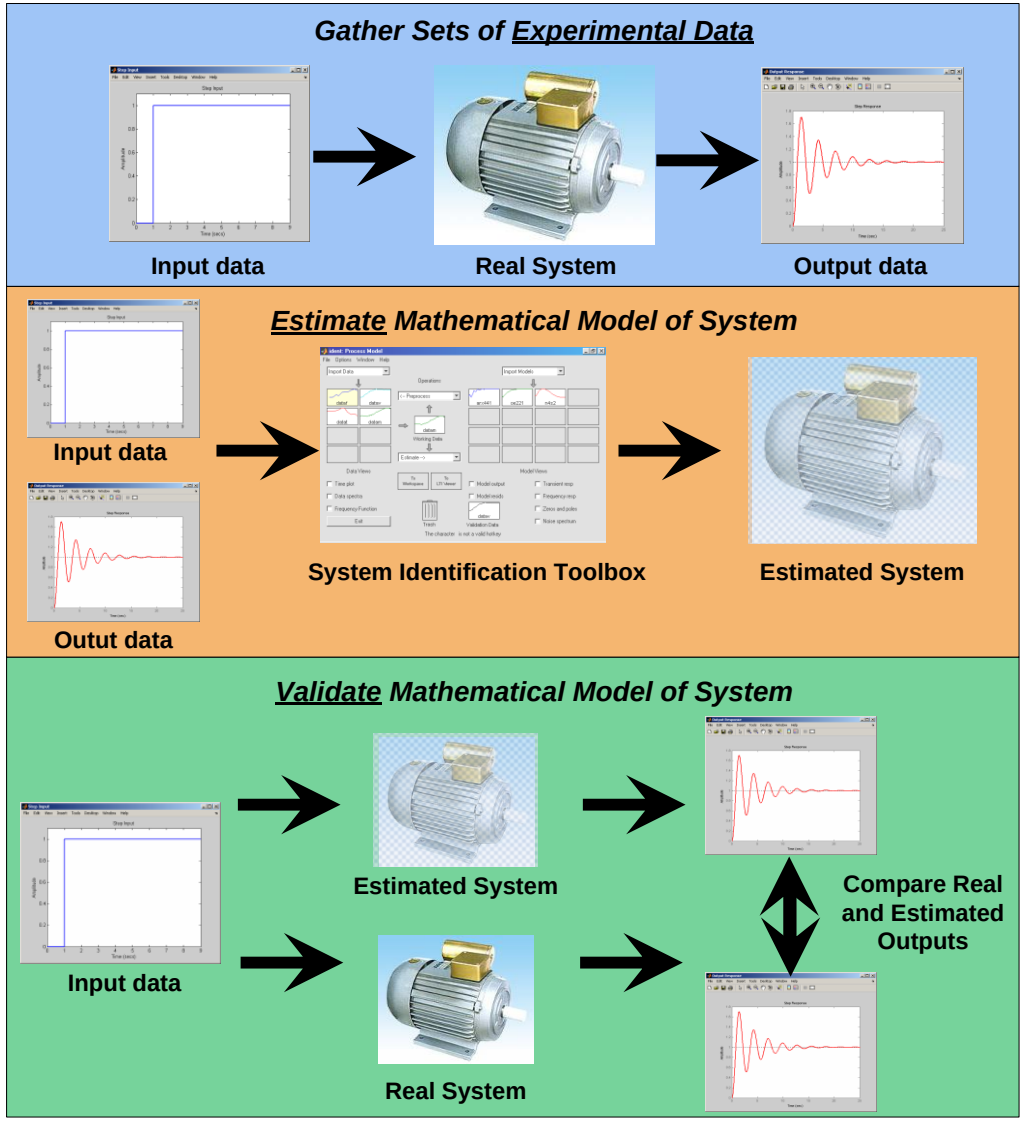
Model

u, y: measured time or frequency domain signals

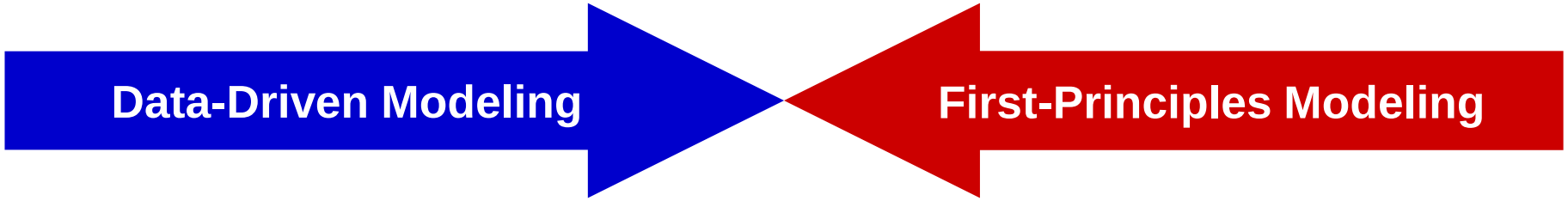


System Identification Process

1. Gather experimental data.
2. Estimate model from data.
3. Validate model with independent data.



Both have advantages & disadvantages



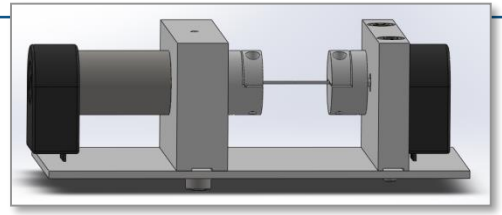
Advantages:
- Fast
- Accurate

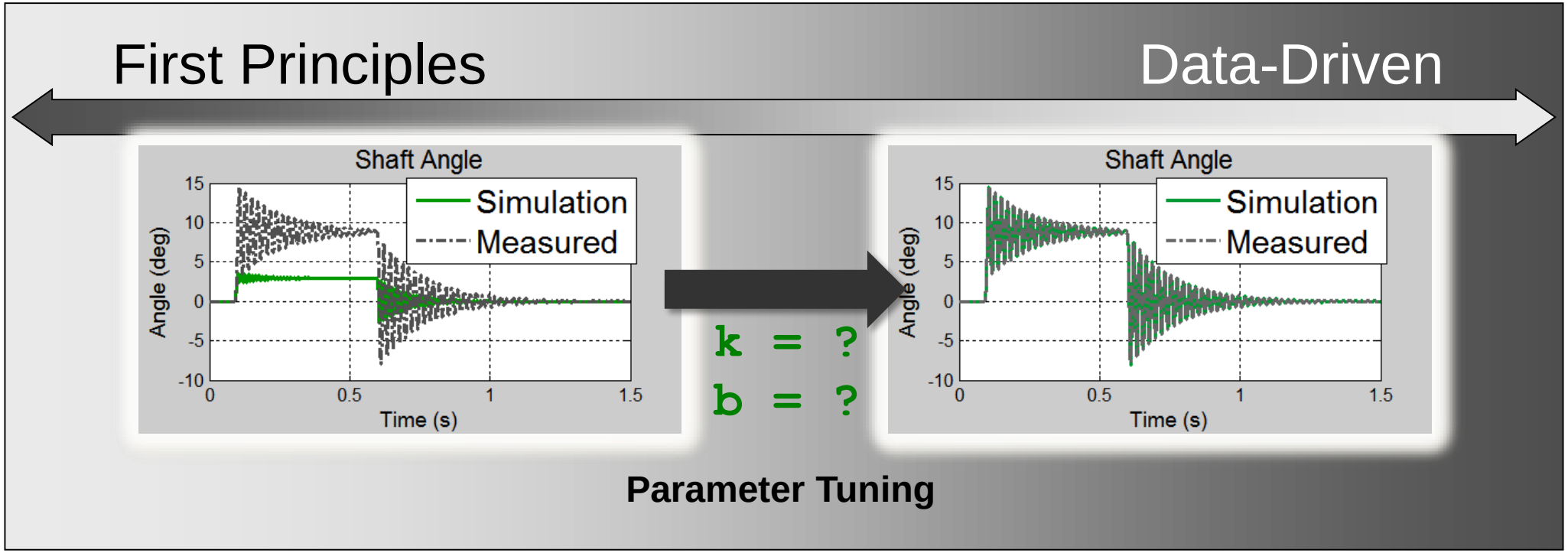
Disadvantages:
- Requires plant
- Requires data acquisition system

Advantages:
- Insight in behavior
- Physical parameters

Disadvantages:
- Time-consuming
- Requires expertise

Modeling Approaches

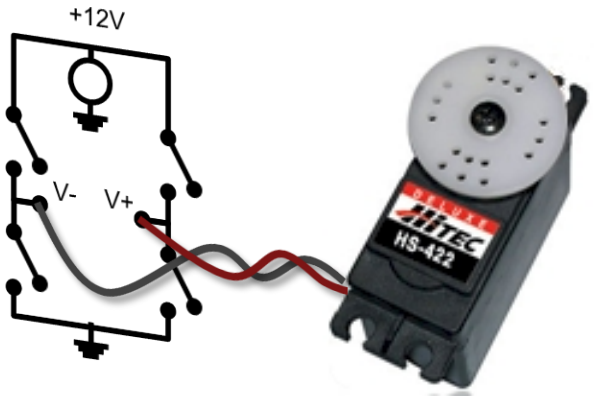




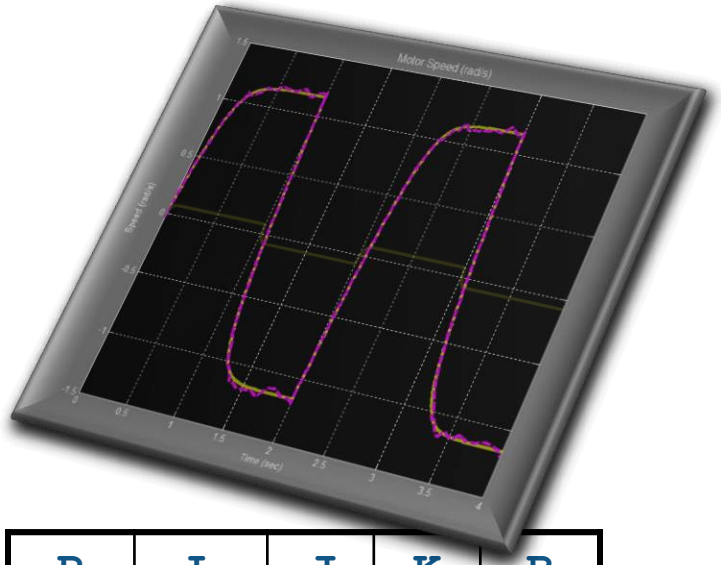
- Purpose: Ensuring parameter values are accurate
- Requirements:
  - Relevant set of measured data is available
  - Physically meaningful parameters can be automatically tuned

web(fullfile(docroot, 'sldo/ref/parameterestimationtool-app.html'))

Validate your models against experimental data using automatic parameter estimation



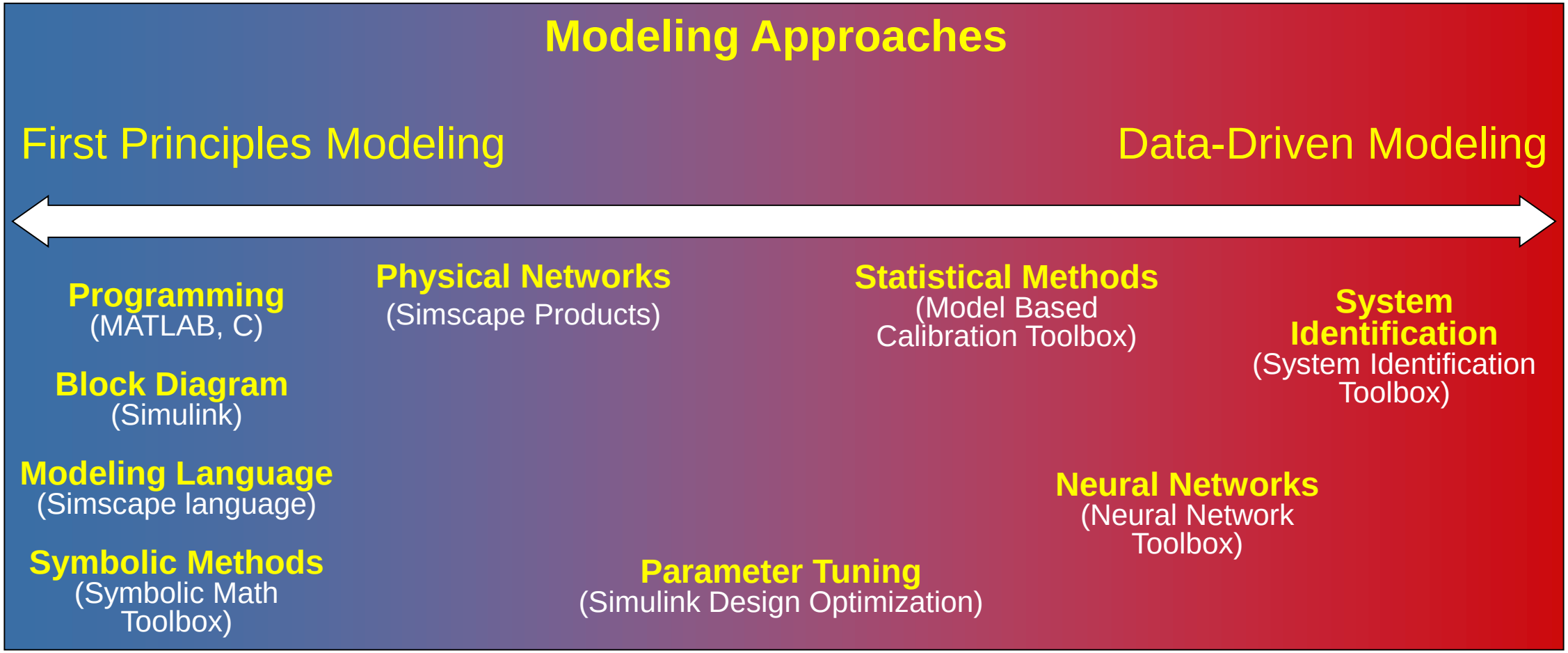
R = Resistance
L = Inductance
J = Inertia
B = Friction
K = Back EMF Constant



R	L	J	K	B
4.03	1e-4	0.11	0.45	1.07

Use ***Simulink Design Optimization*** to tune model parameters by taking advantage of the capabilities in the ***Optimization Toolbox*** directly in Simulink

Modeling Physical Systems With MathWorks Products



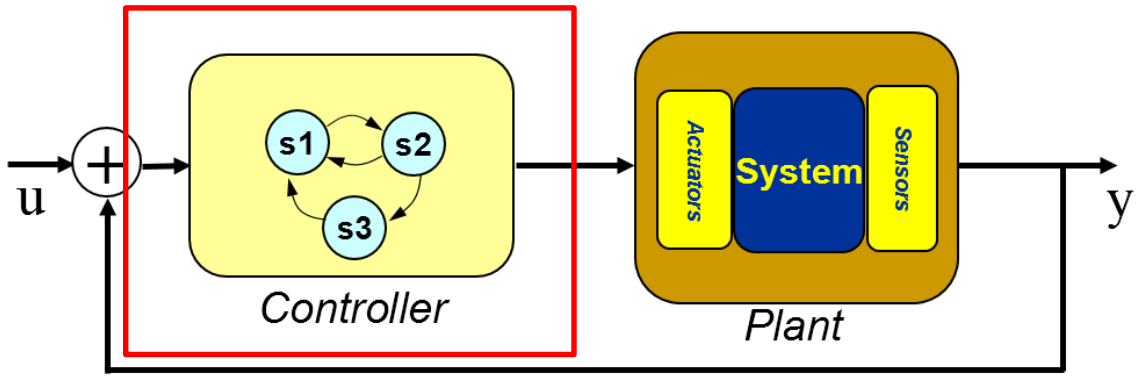
Control Design Process

- **Modeling:**

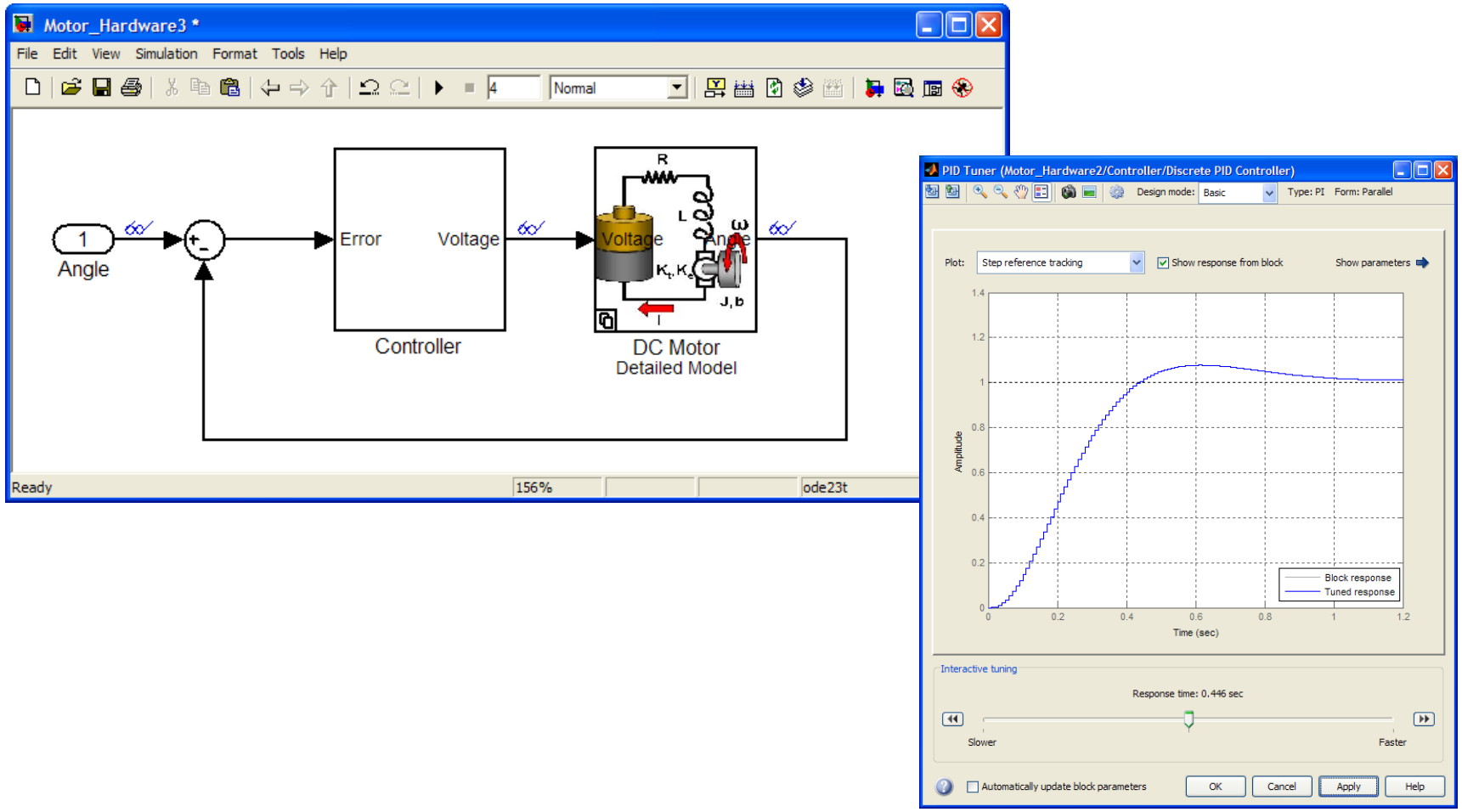
  understanding your design problem

- **Control Design:**

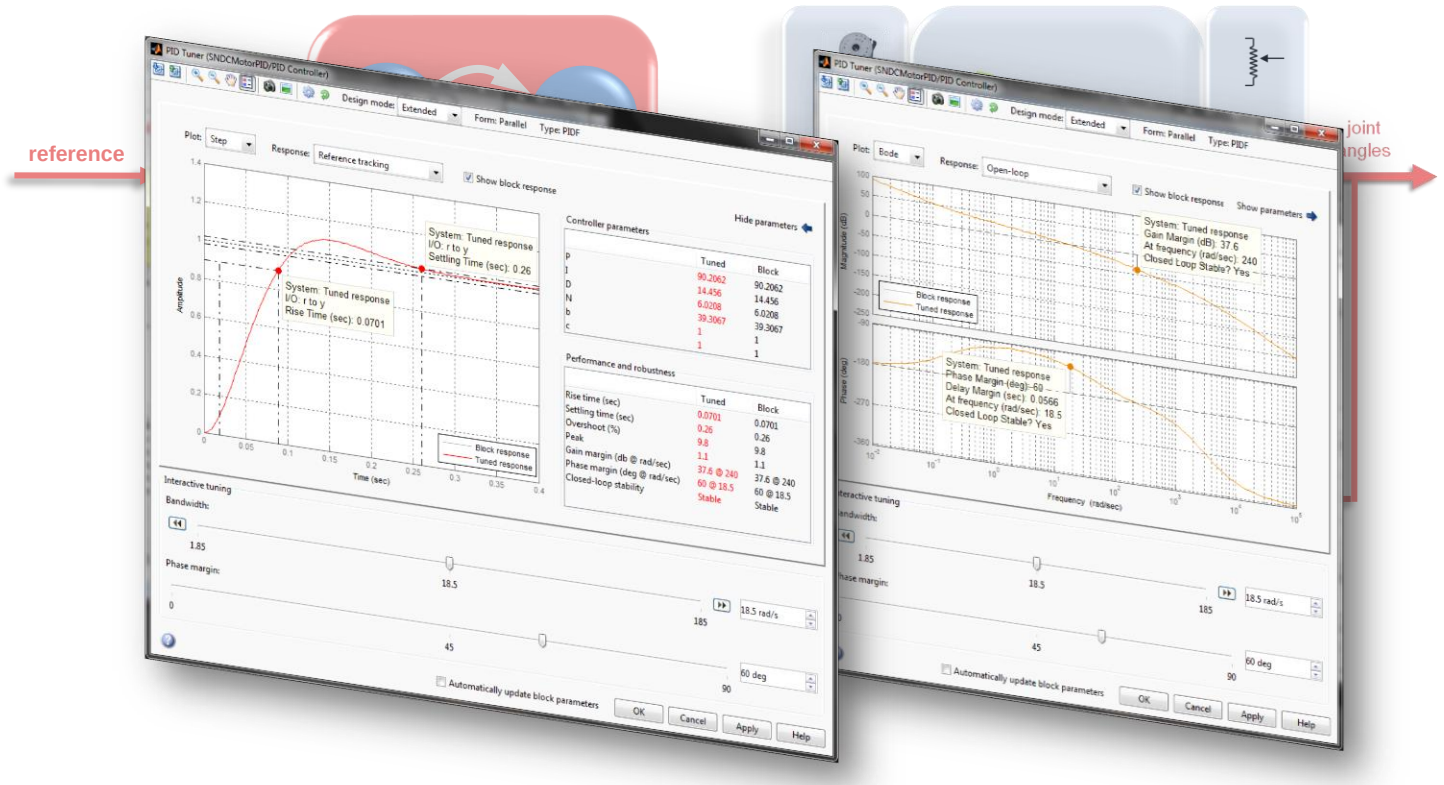
  solving your design problem



DC Motor Controller Design
PID block with Simulink Control Design



Automatic PID tuning

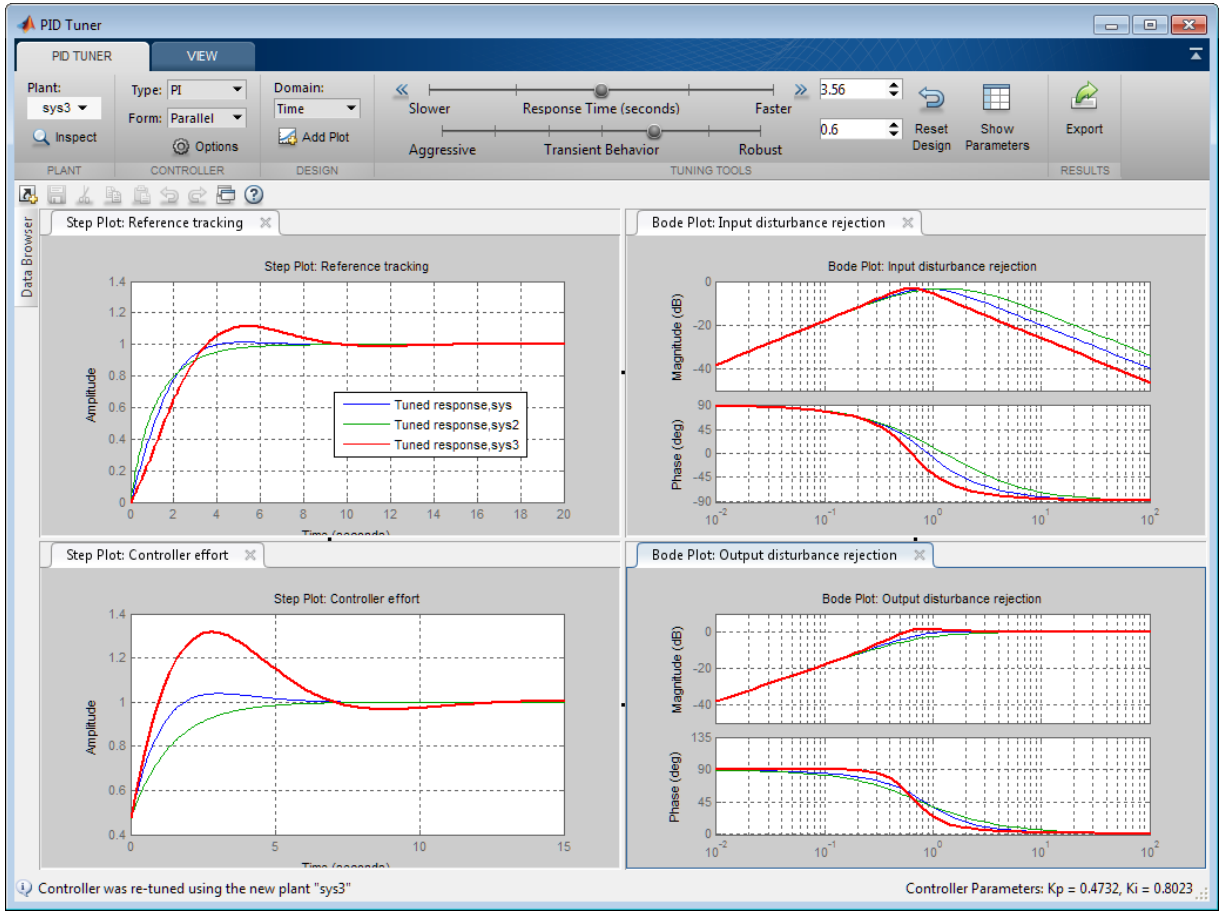


Use ***Simulink Control Design*** and the ***Control System Toolbox*** to automatically linearize the plant, design and tune your PID controllers

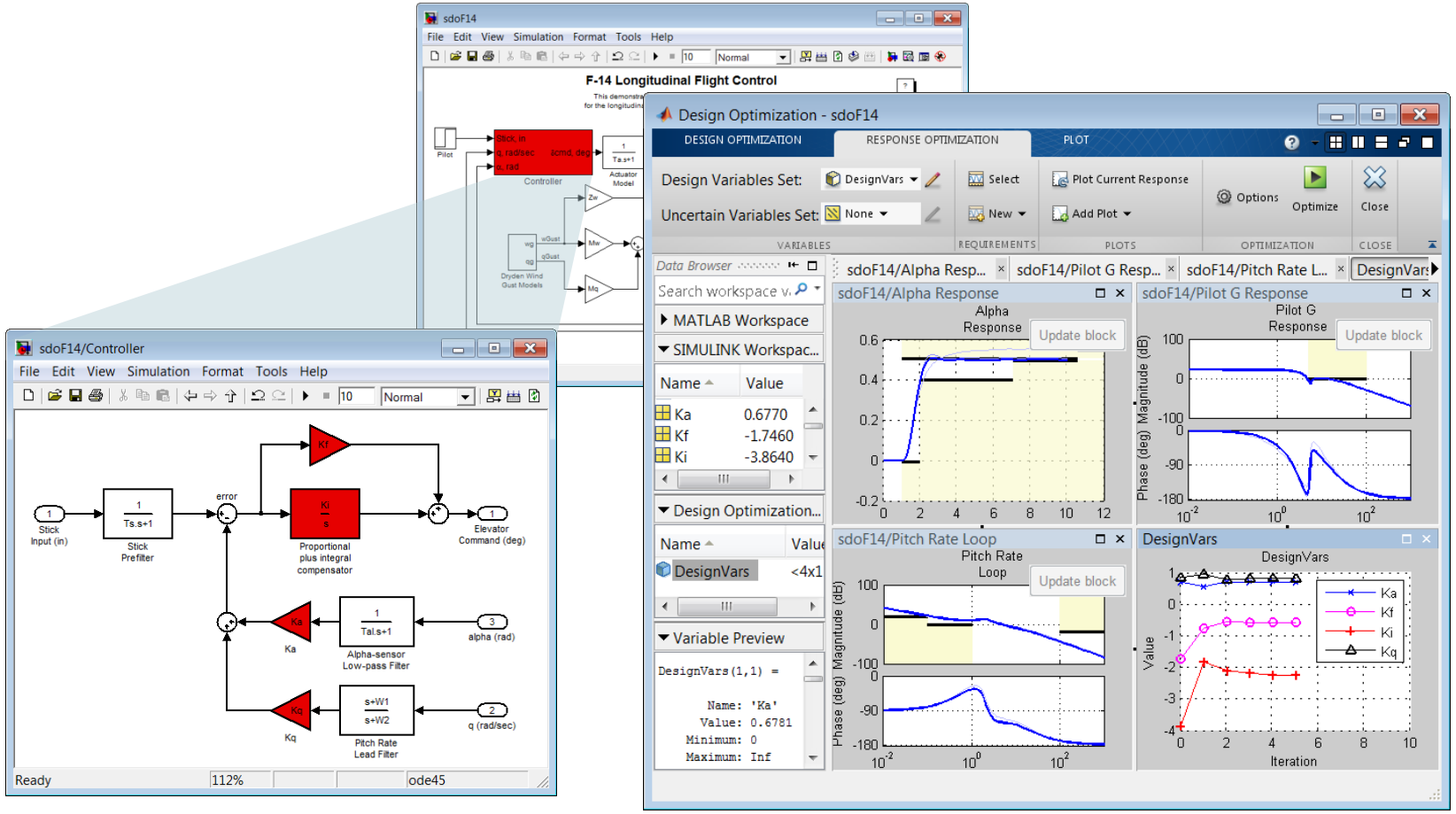
PID Tuner App

Interactive tuning of PID controllers

- Automatically finds the design that balances performance and robustness
- Lets you easily try different controller structures
- Provides two sliders for fine-tuning the design
- Several response plots can be displayed simultaneously

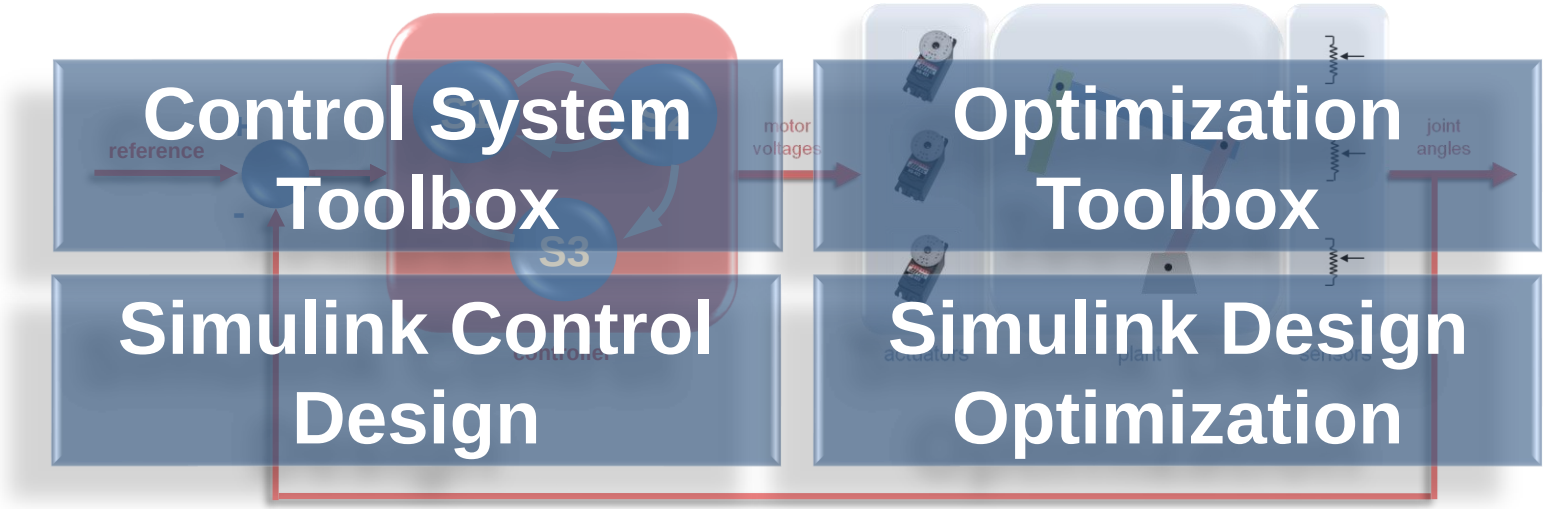


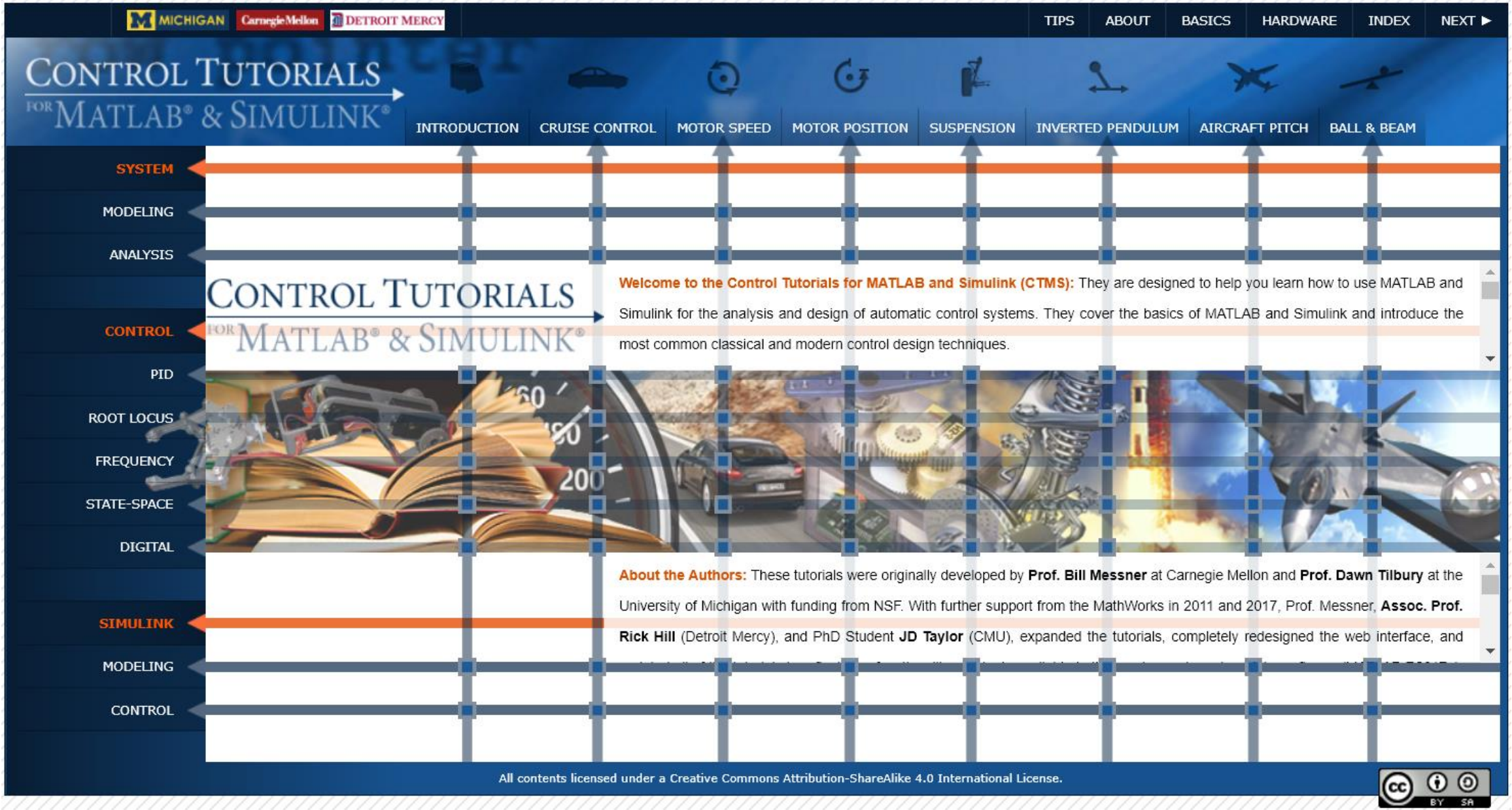
Fine tune controller gains using response optimization



Use ***Simulink Design Optimization*** to optimize overall system response against requirements in the time and frequency domain

Base/recommended control design tools





http://ctms.engin.umich.edu/CTMS/index.php?aux=Home

Offline Control Tutorials for ML & SL

https://www.mathworks.com/academia/courseware/control-tutorials/

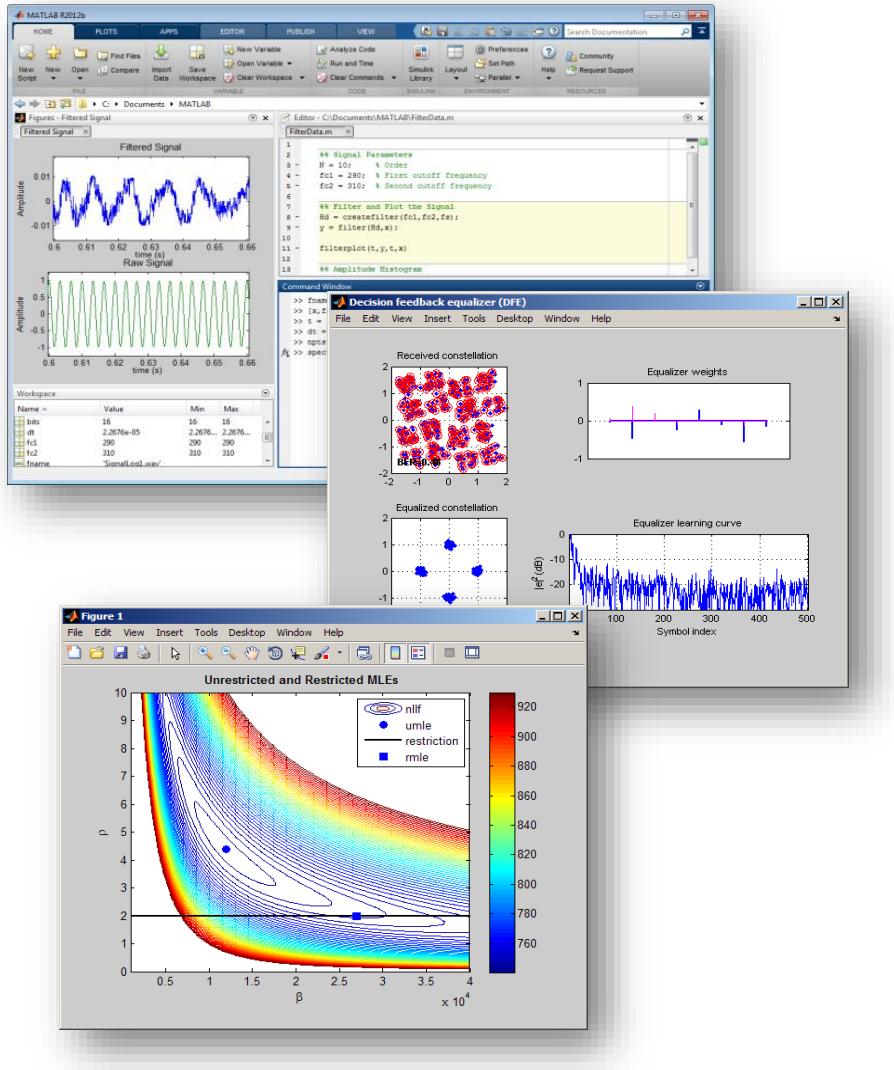
Introduction to MATLAB®

Agenda

- What is MATLAB?
- Getting started with MATLAB
- Data Generation
- Data Indexing
- Calculation
- Visualization
- Programming Keywords
- Script
- Function

What is MATLAB?

- The leading environment for technical computing
- High-level language
  - Native support for vector and matrix operations
  - Built-in math and visualization functions
- Interactive development environment
  - Interactive and easy to get started
  - Ideal for iterative exploration and design
- Used for:
  - Numerical computation
  - Data analysis and visualization
  - Algorithm development and programming
  - Application development and deployment



MATLAB

is the **easiest** and

most **productive** environment

for **engineers** and **scientists**

How do you feel to learn programming language?

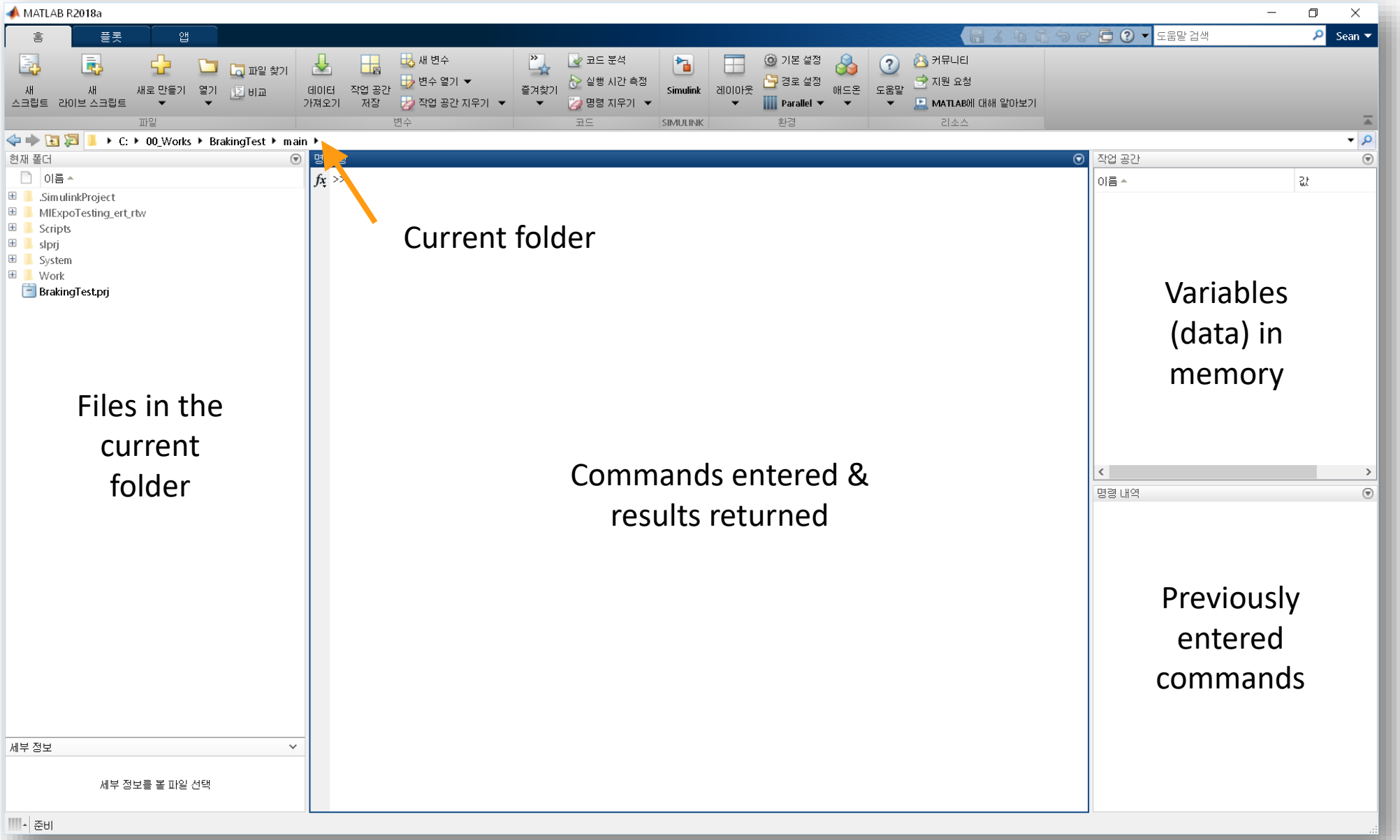
How to Make Program about Drawing Sine Wave

```
void mysin(double y[63])
{
  int i0;
  static const double x[63] = { 0.0, 0.1, 0.2, 0.30000000000000004, 0.4, 0.5,
    0.60000000000000009, 0.70000000000000007, 0.8, 0.9, 1.0, 1.1,
    1.2000000000000002, 1.3, 1.4000000000000001, 1.5, 1.6, 1.7000000000000002,
    1.8, 1.9000000000000001, 2.0, 2.1, 2.2, 2.3000000000000003,
    2.4000000000000004, 2.5, 2.6, 2.7, 2.8000000000000003, 2.9000000000000004,
    3.0, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0, 4.1, 4.2, 4.3, 4.4,
    4.5, 4.6, 4.7, 4.8, 4.9, 5.0, 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9,
    6.0, 6.1000000000000005, 6.2 };

  for (i0 = 0; i0 < 63; i0++) {
    y[i0] = sin(x[i0]);
  }
}
```

Graph?

```
>> t=0:0.1:2*pi;
>> y=sin(t);
>> plot(t,y)
fx >>
```

The MATLAB® Desktop

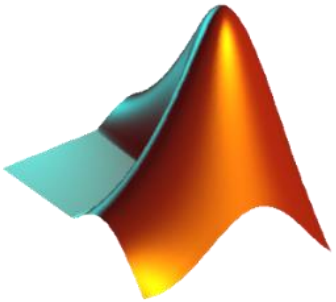


Data Generation

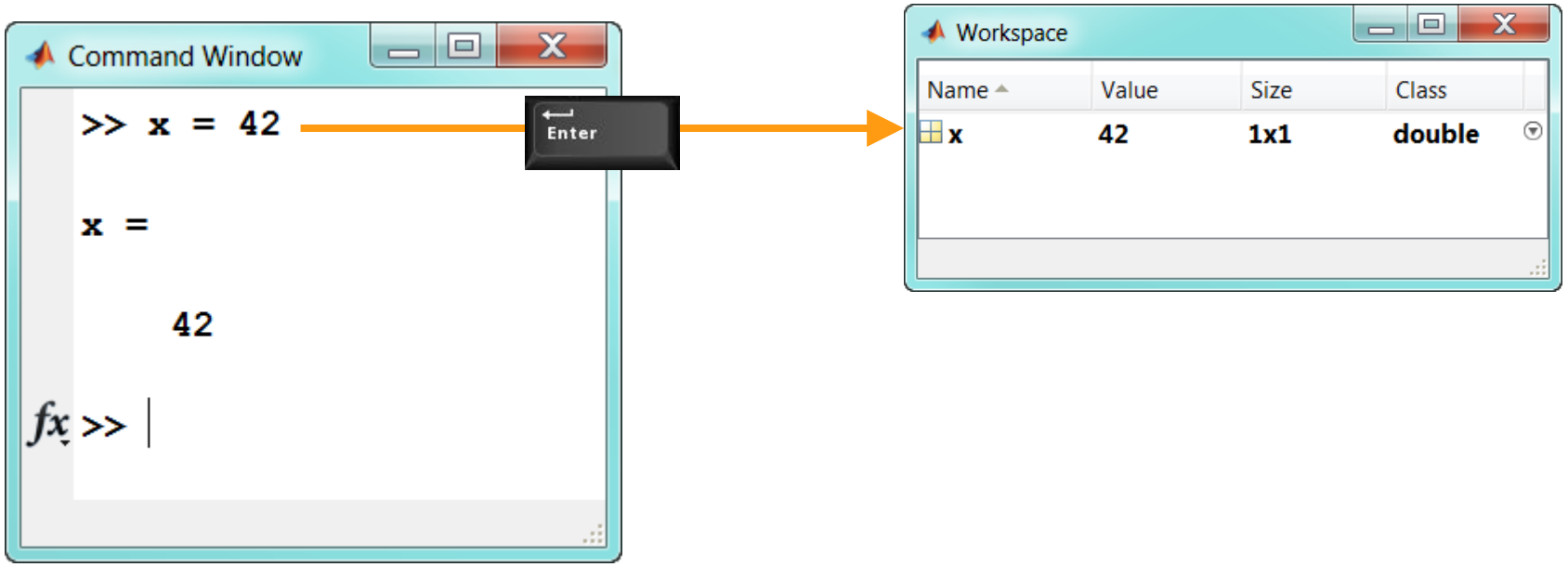
Generating Data in MATLAB®

Generating data
from external source

Generating data
from user in ML



Creating Numeric Data from Commands



>> variable_name1=number or calculable equations;

Creating Scalar, Vectors, and Matrices

1. Scalar

```
>> variable_name=num; or  variable_name=[num];
```

3.1

2. Vector

```
>> variable_name=[num1; num2];
>> variable_name=[num1, num2];
>> variable_name=a:dx:b;  or  variable_name=linspace(a,b,n);
```

3.1
4.2
9.0

6	8	10	12	14	16

3. Matrix

```
>> variable_name=[num1, num2; num3 num4];
>> variable_name=function(num1,num2);
```

1	0	0
0	1	0
0	0	1

Concatenating Arrays

A

2	3
5	7
11	13

B

-1	1
1	-1

C

0
8
0

```
>> X = [A;B]
```

2	3
5	7
11	13
-1	1
1	-1

```
>> Y = [A,C]
```

2	3	0
5	7	8
11	13	0

```
>> Y = cat(2, A, C)
```

2	3	0
5	7	8
11	13	0

Importing Supported File Data from External

>> docsearch('Supported File Formats for Import and Export')

Audio

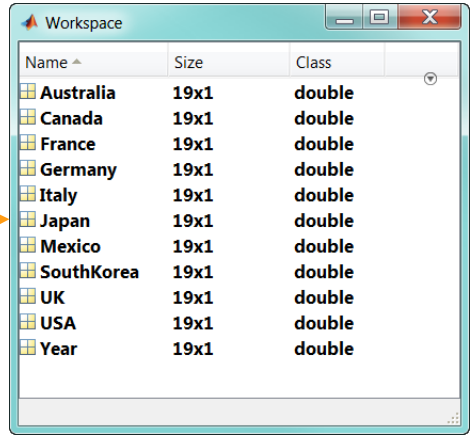
Excel

Text

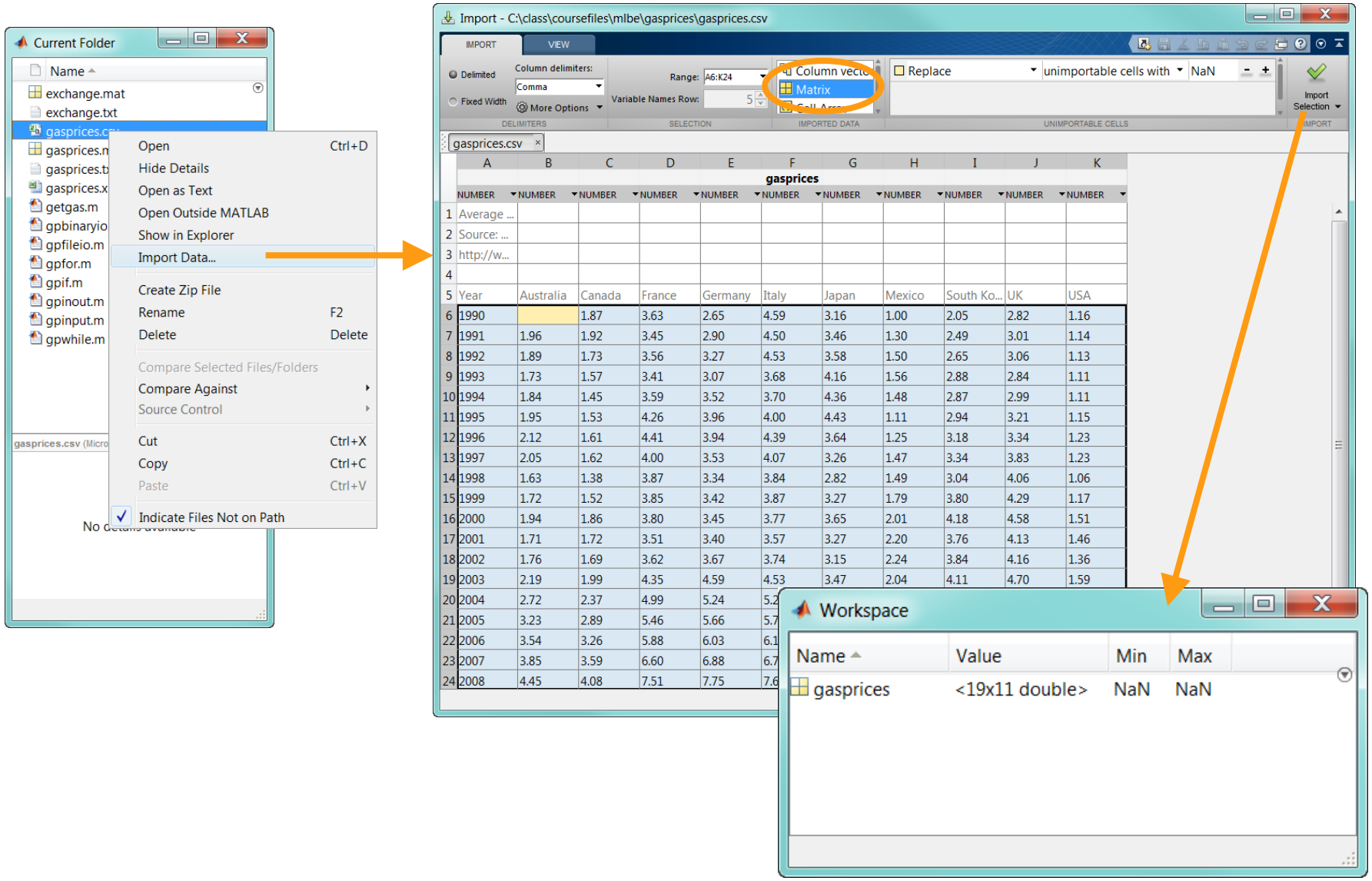
Video

…

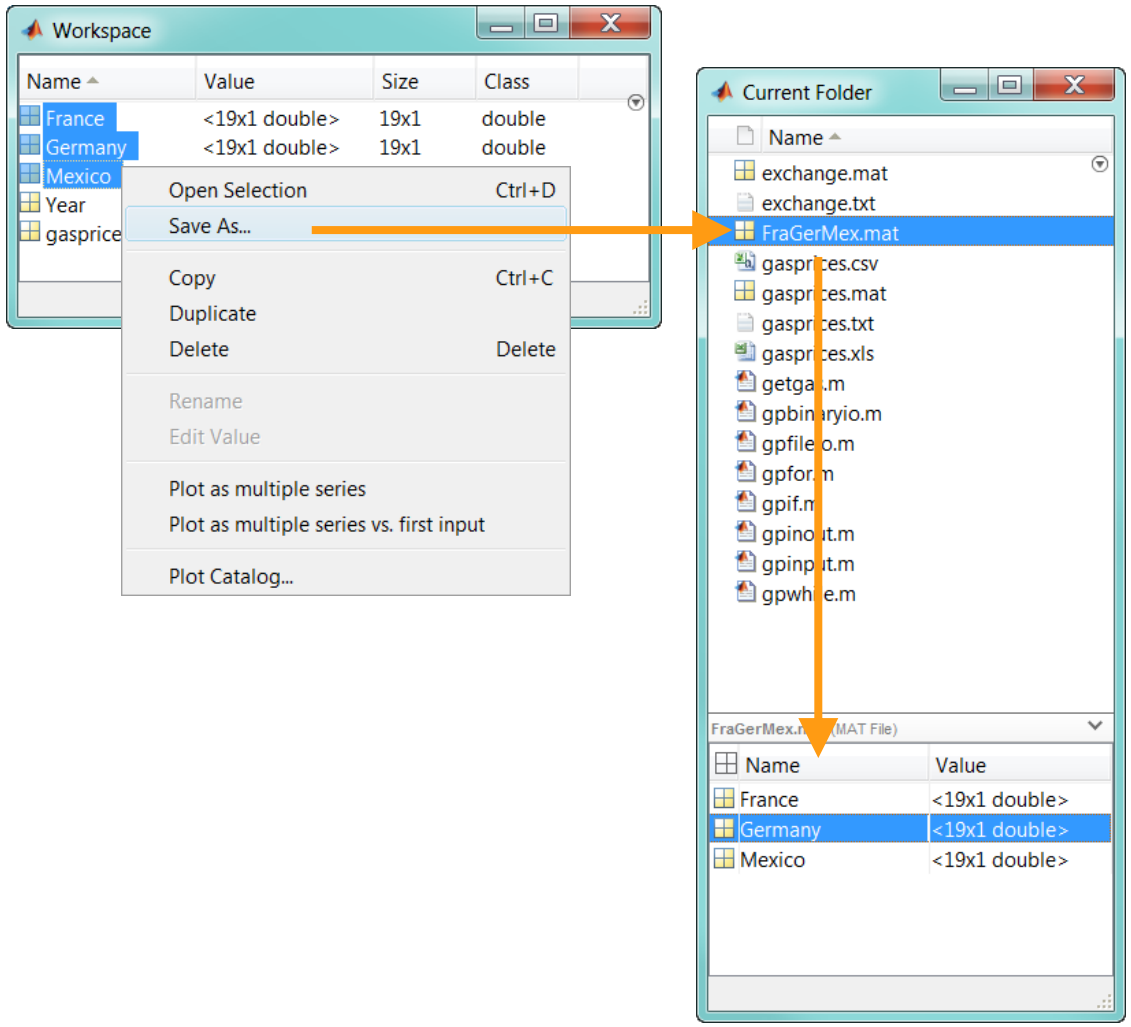
```
data = xyzread('myfile')

Ex) audioread()
    xlsread()
    dlmread()
    imread()
```



Interactive Importing



Saving and Loading Variables



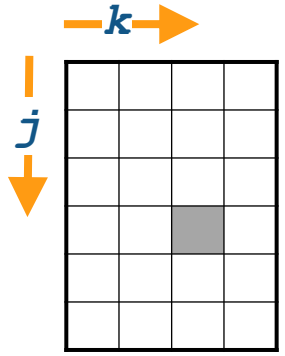

```
>> save filename variable_name1 variable_name2
>> load filename.mat
```

Data Indexing

Row-Column Indexing

```
>> A = [1,2;3,4];
A =
     1 (1,1)  2 (1,2)
     3 (2,1)  4 (2,2)

>> X = A(1 Row Indices ,2 Column Indices )
X = 2

>> Y = A(1:2,2)
Y = 2
    4

>> Z = A(end,:)
Z = 3   4
```

A(*j*,*k*)



Linear Indexing

```
>> A = magic(5)
A =
     1 17     6 24    11  1    16  8    21 15
     2 23     7  5    12  7    17 14    22 16
     3  4     8  6    13 13    18 20    23 22
     4 10     9 12    14 19    19 21    24  3
     5 11    10 18    15 25    20  2    25  9
                                      end
```

Indices Data

```
>> x = A(20)
```

Calculation

Array Operations

1. +, -, *, ^, / (A/B=A*inv(B)), \ (A\b=inv(A)*B)

2. .*, .^, ./ (A elements/B elements), .\ (B elements\A elements)

```
>> C = A + B
```

$$\begin{bmatrix} 3 \\ 5 \\ 4 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

```
>> C = A + B
```

$$\begin{bmatrix} 4 \\ 5 \\ 3 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} + \begin{bmatrix} 2 \end{bmatrix}$$

```
>> C = A * B
```

Error = $\begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$

```
>> C = A \ B
```

$$\begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 5 \end{bmatrix} \backslash \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

```
>> C = A .* B
```

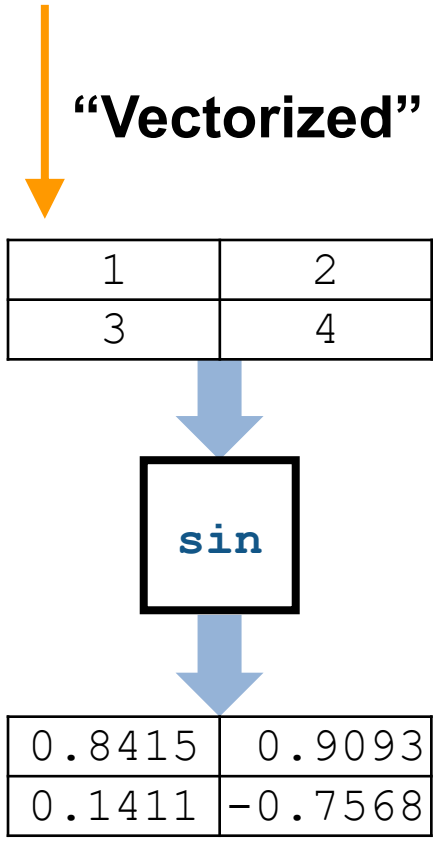
$$\begin{bmatrix} 2 \\ 6 \\ 3 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} .\times \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

```
>> C = A ./ B
```

$$\begin{bmatrix} 2 \\ 1.5 \\ 0.33 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} ./ \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

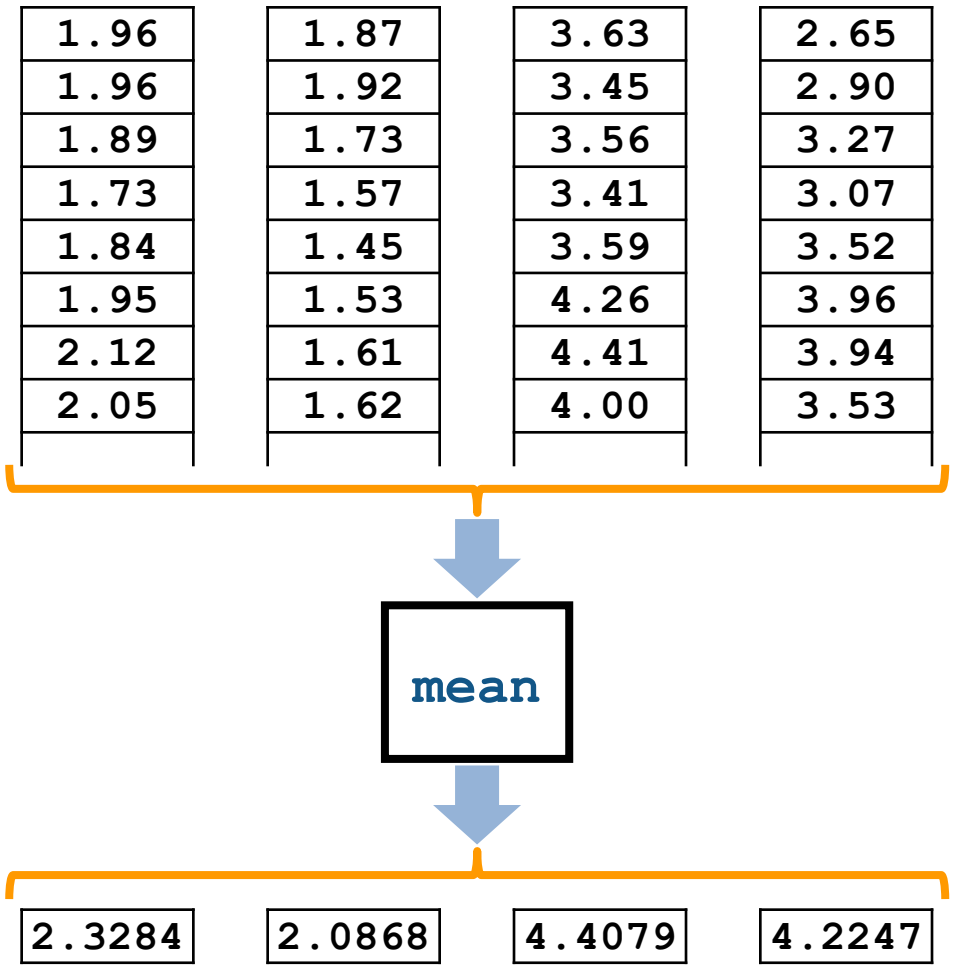
Mathematical functions

```
>> B = sin(A);
```




```
sin
sind
sinh
asin
exp
log
log2
log10
sqrt
nthroot
abs
angle
floor
ceil
round
mod
```

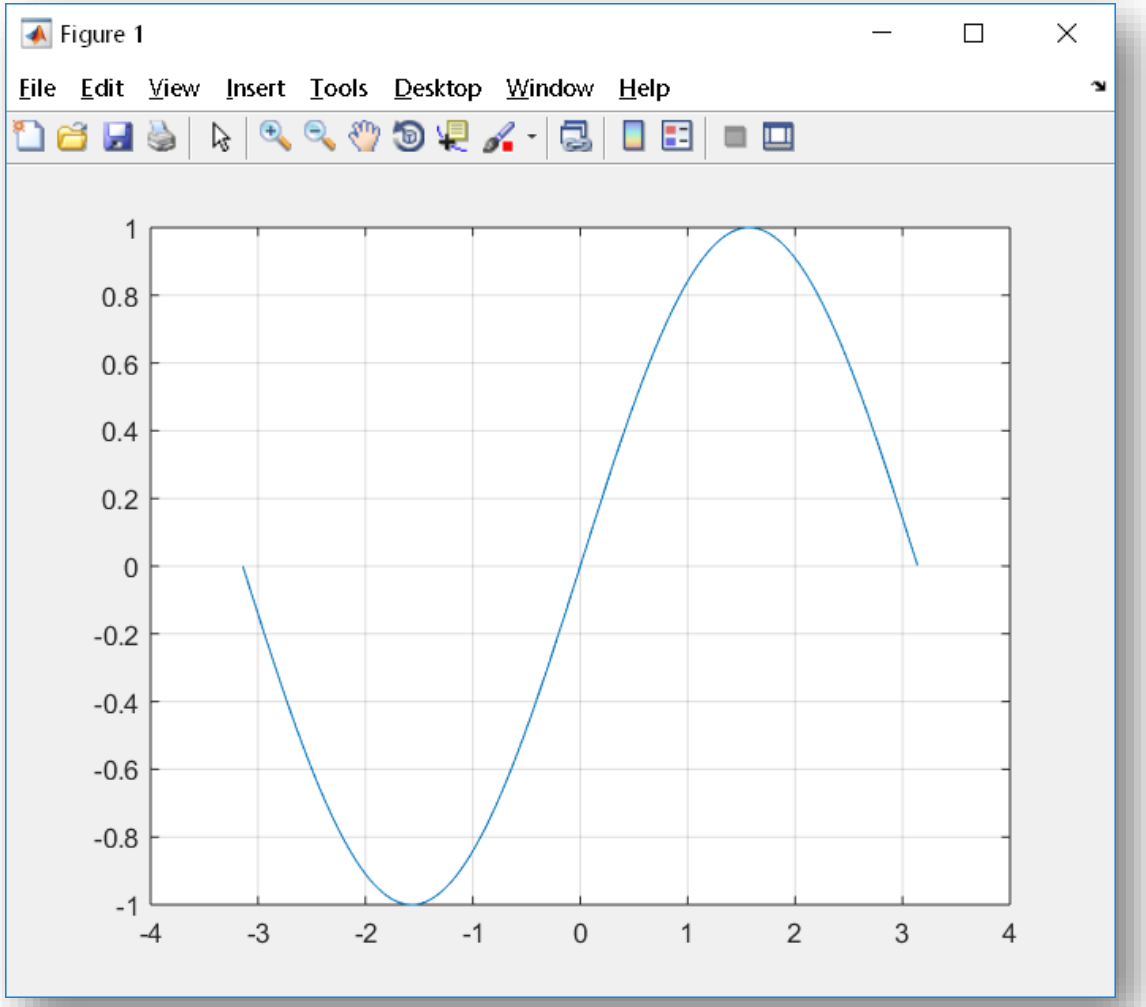
Statistical functions



- max
- min
- mean
- median
- std
- sum
- prod
- diff
- gradient
- cumsum
- cumprod
- corrcoef
- cov

Visualization

Plot Function



Plot Sine Function

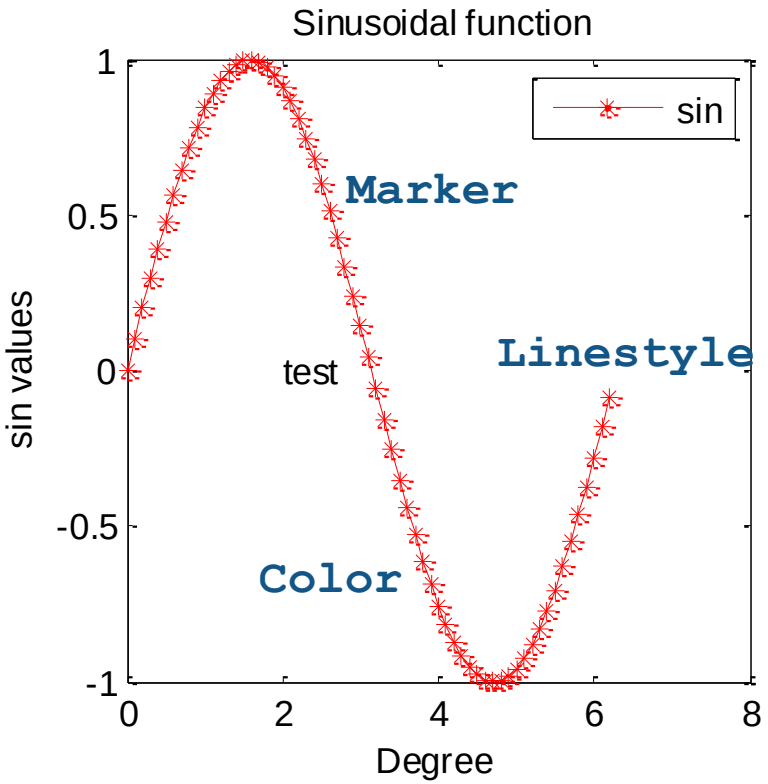
Plot the sine function over the domain $-\pi \leq x \leq \pi$

```
x = -pi:0.01:pi;
y = sin(x);
plot(x,y), grid on
```

Plot Options

Color		Marker		Line Style	
`b`	blue	`.`	point	`-`	solid
`g`	green	`o`	circle	`--`	dashed
`r`	red	`x`	(diagonal) cross	`:`	dotted
`c`	cyan	`+`	(vertical) cross	`-.`	dash-dot
`m`	magenta	`*`	star		
`y`	yellow	`s`	square		
`k`	black	`d`	diamond		
`w`	white	`v`	triangle (down)		
		`^`	triangle (up)		
		`<`	triangle (left)		
		`>`	triangle (right)		
		`p`	pentagram		
		`h`	hexagram		

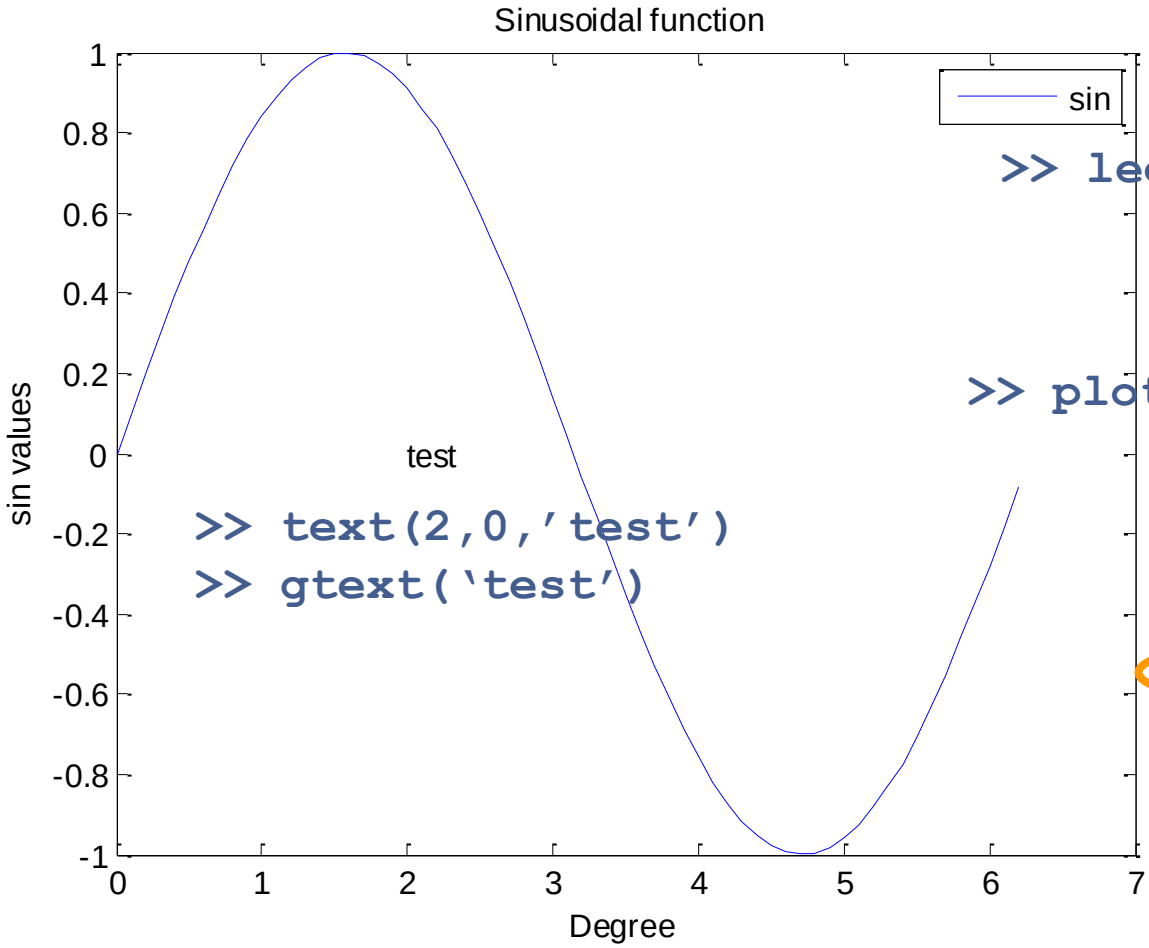
```
>> plot(x,y,'PropertyName',Value)
```




```
>> plot(x,y,'r*-')
```

Annotating Plots

```
>> title(Sinusoidal function')
```




```
>> ylabel('sin values')
```

```
>> legend('sin')
```

```
>> plot( x   ,  y )
```

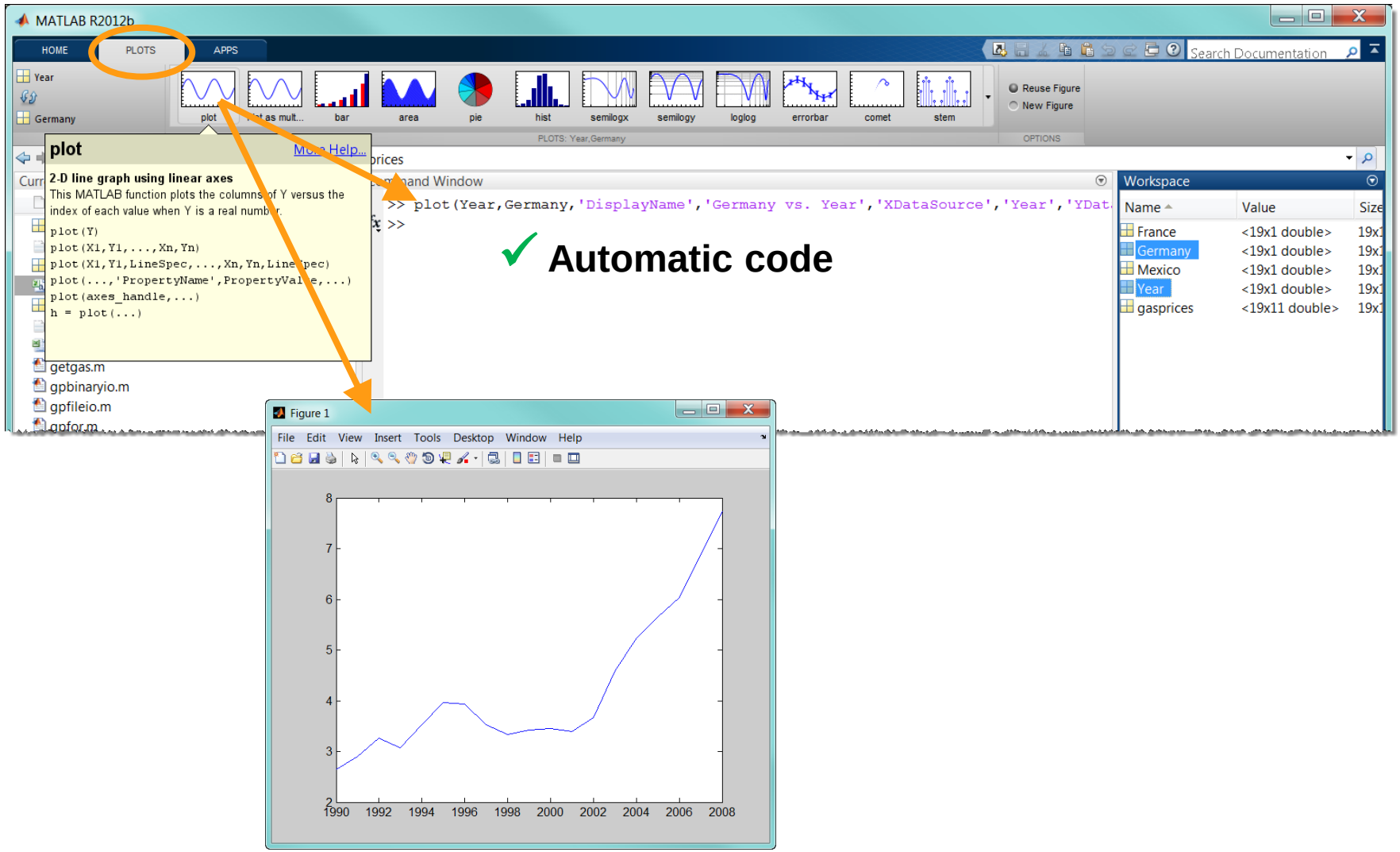
0	0
0.1	0.1
0.2	0.2
…	…
6.2	0

```
>> text(2,0,'test')
>> gtext('test')
```

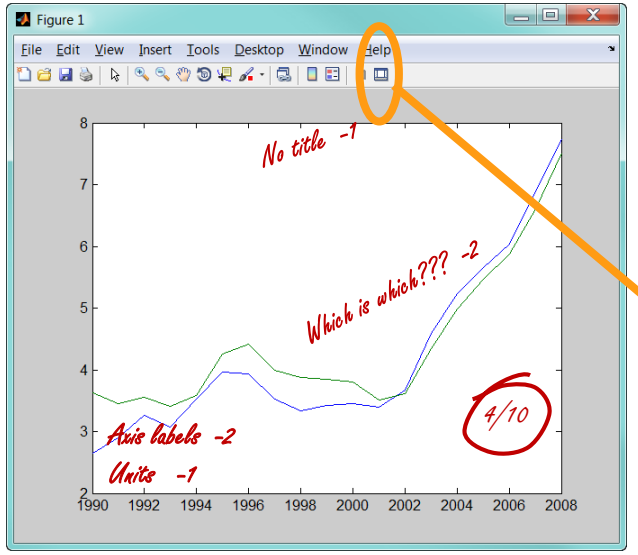
```
>> xlabel('Degree')
```

web(fullfile(docroot, 'matlab/ref/plot.html#btzitot-4'))

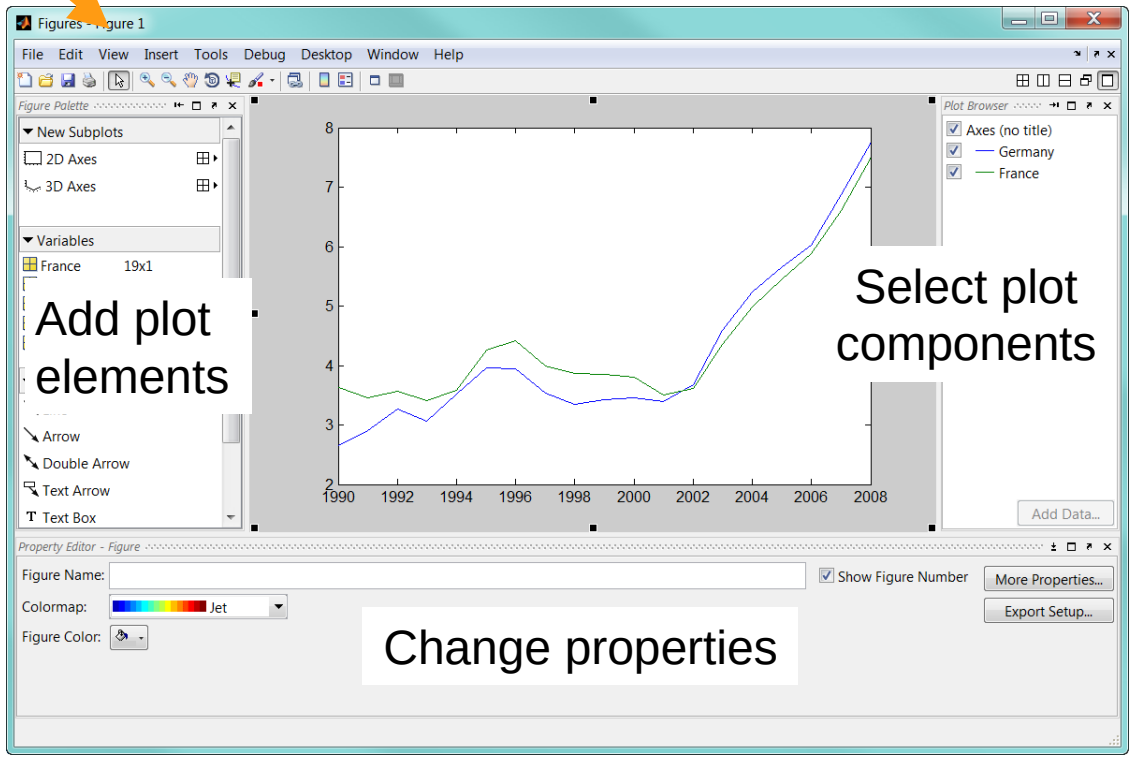
Plotting the Data



Plot Tools

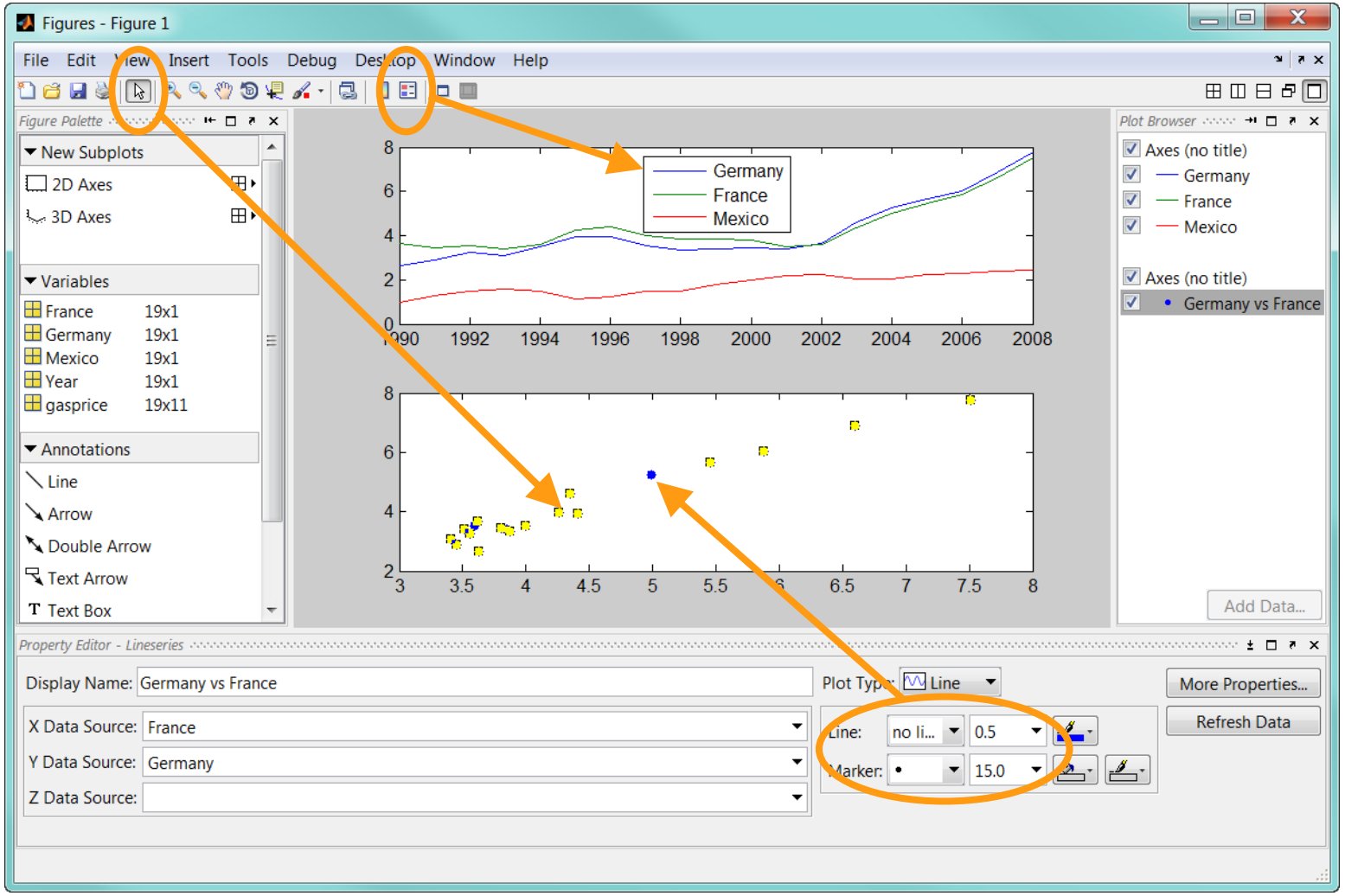



```
>> x = linspace(1,10,25);
>> y = sin(x);
>> plot(x,y)
>> plottools
```



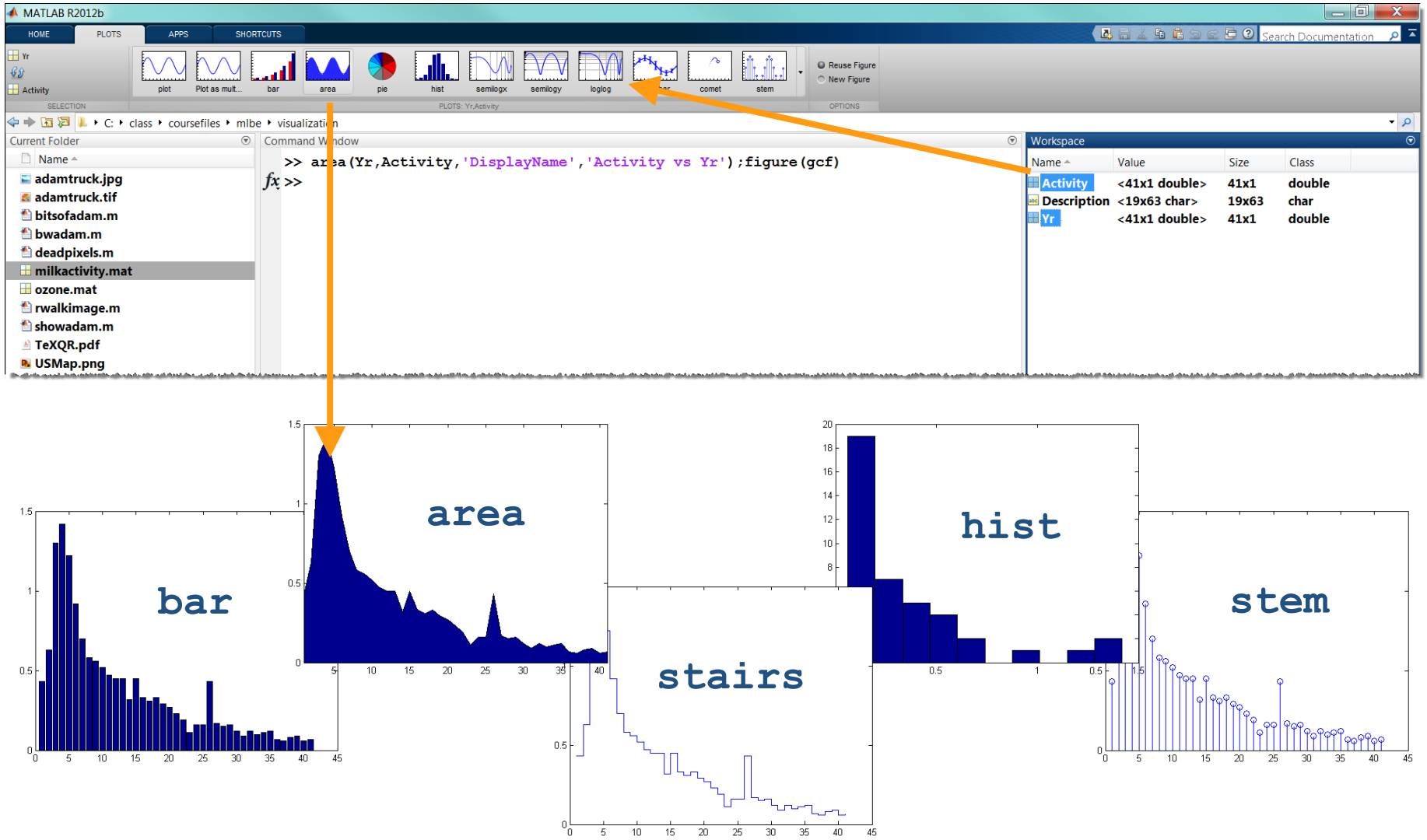
web(fullfile(docroot, 'matlab/ref/plottools.html'))

Formatting the Plot



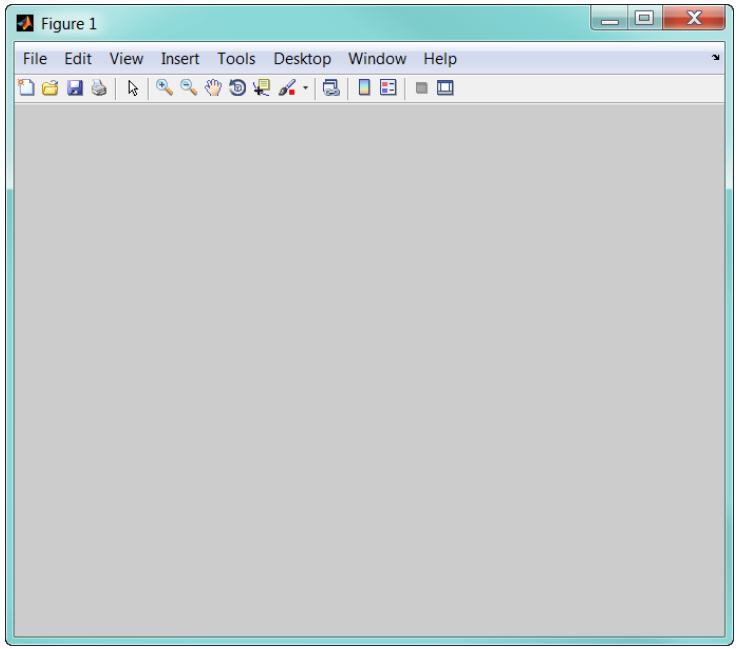
web(fullfile(docroot, 'matlab/creating_plots/plotting-tools-interactive-plotting.html'))

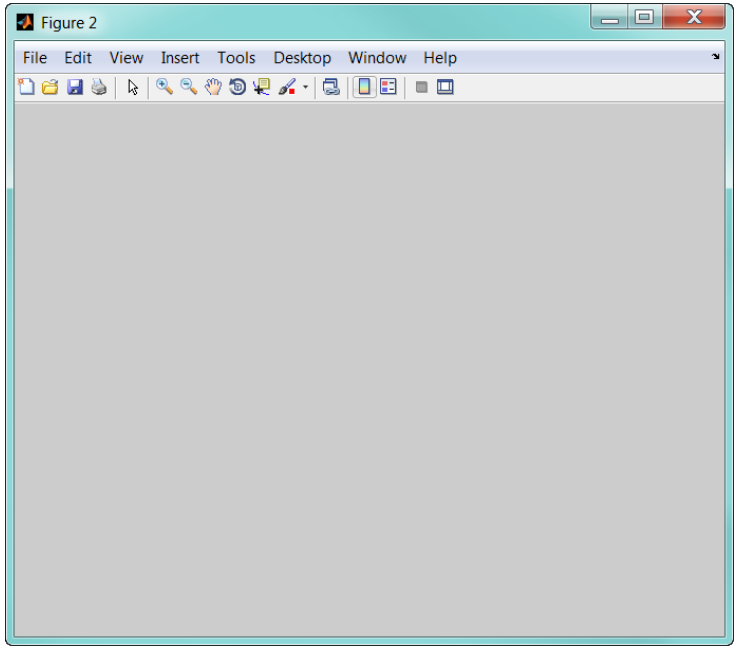
Additional Vector Plot Types



Multiple Figures

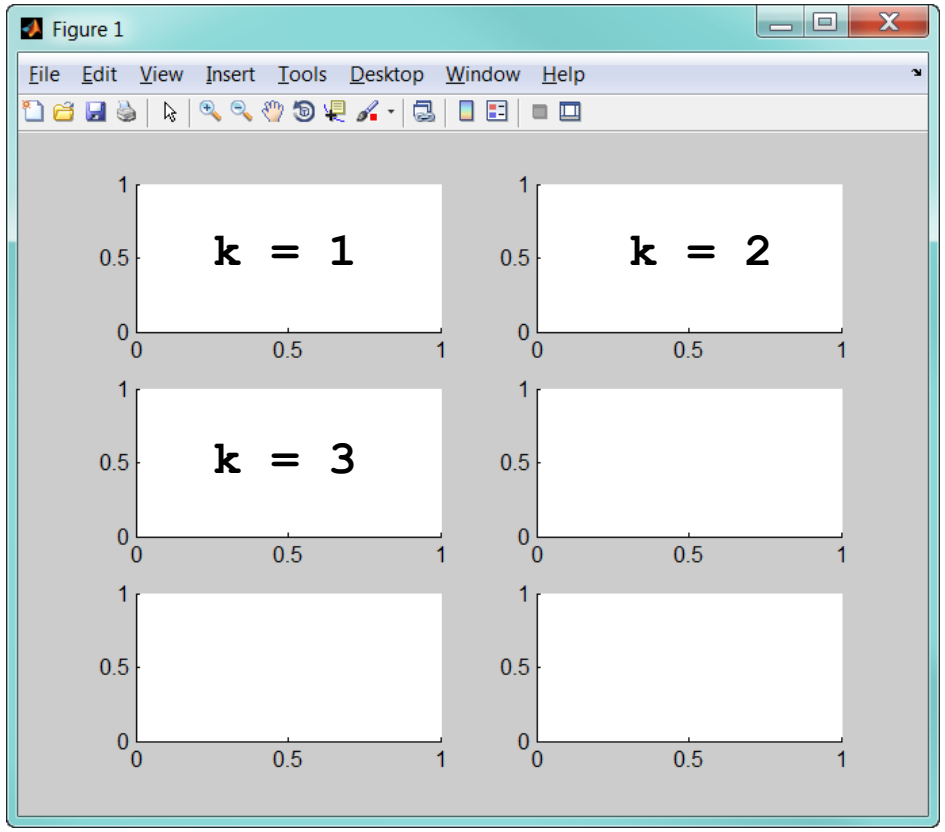
```
>> figure
>> figure(n)
```





Multiple Axes

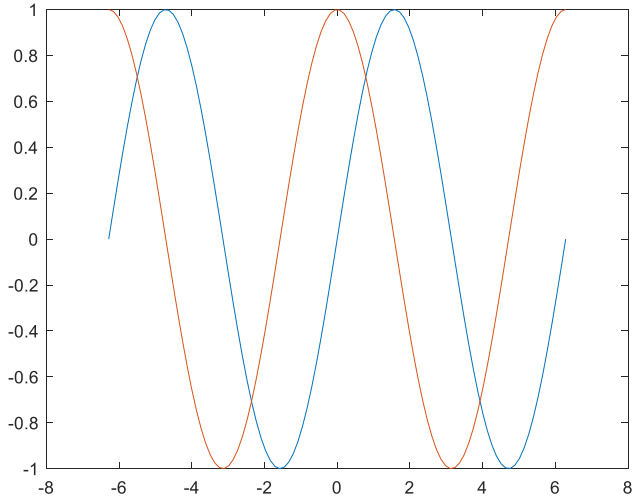
```
>> subplot(3,2,k)
```



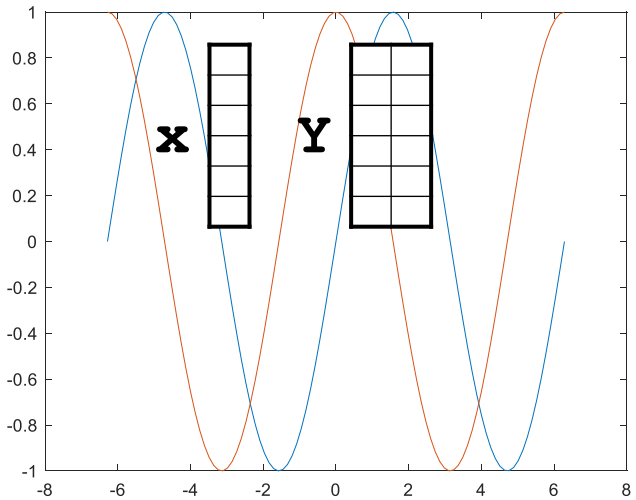
Multiple Plots

Plot Multiple Lines

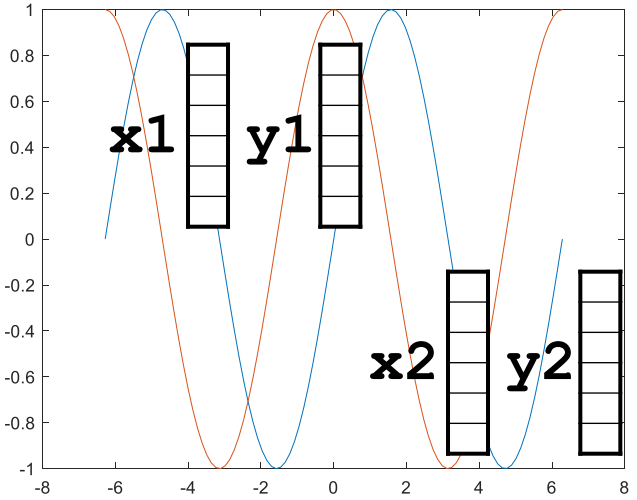
```
x = linspace(-2*pi,2*pi);
y1 = sin(x);
y2 = cos(x);
```




```
>> plot(x,y1)
>> hold on;
>> plot(x,y2)
```

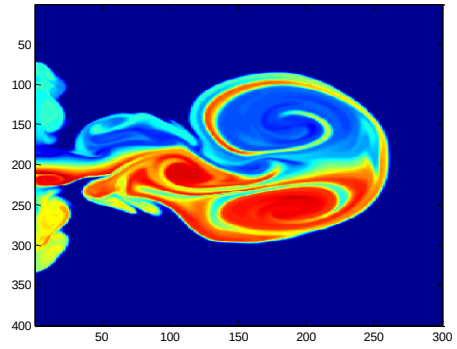



```
>> plot(x,[y1;y2])
```



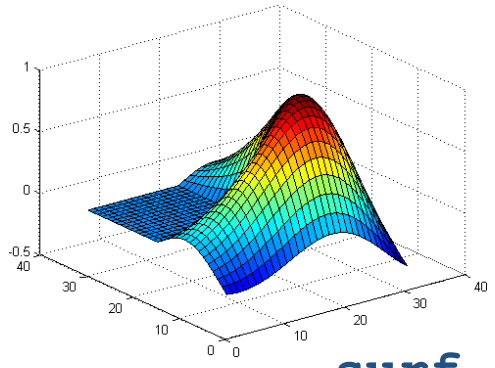

```
>> plot(x,y1,x,y2)
```

Additional Visualization



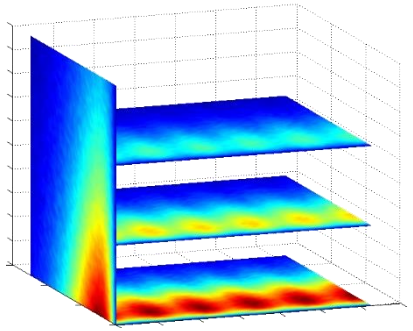
imagesc

web(fullfile(docroot, 'matlab/ref/imagesc.html'))



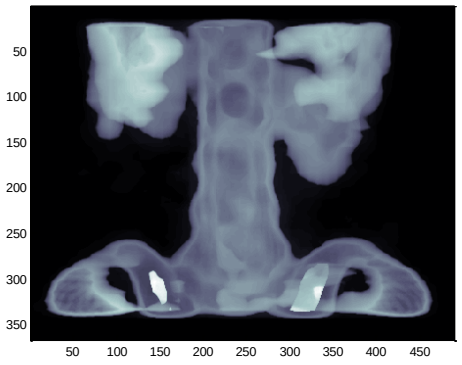
surf

web(fullfile(docroot, 'matlab/ref/surf.html'))



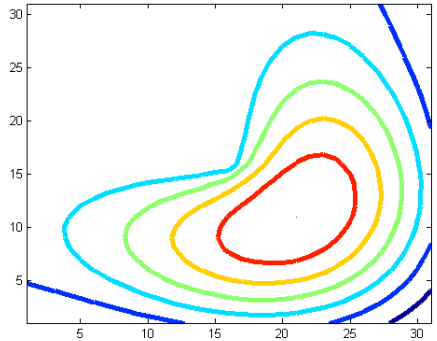
slice

web(fullfile(docroot, 'matlab/ref/slice.html'))



colormap

web(fullfile(docroot, 'matlab/ref/colormap.html'))



contour

web(fullfile(docroot, 'matlab/ref/contour.html'))

Programming Keywords

Programming Keywords

```
for index= first:increment:last
    statements
end
```

```
>> a=0;
 for i=1:10
      a=a+i;
 end
 disp(a)
```

Programming Keywords

```
while condition
    statements
end
```

```
>> a=0; i=1;
   while i<11
       a=a+i;
       i=i+1;
   end
   disp(a)
```

Programming Keywords

```
if logical test1
    statement1
elseif logical test2
    statement2
else
    statement3
end
```

```
>> a=0; b=0; i=1;
while i<11
if mod(i,2)==1
a=a+i;
else
b=b+i;
end
i=i+1;
end
disp([a b])
```

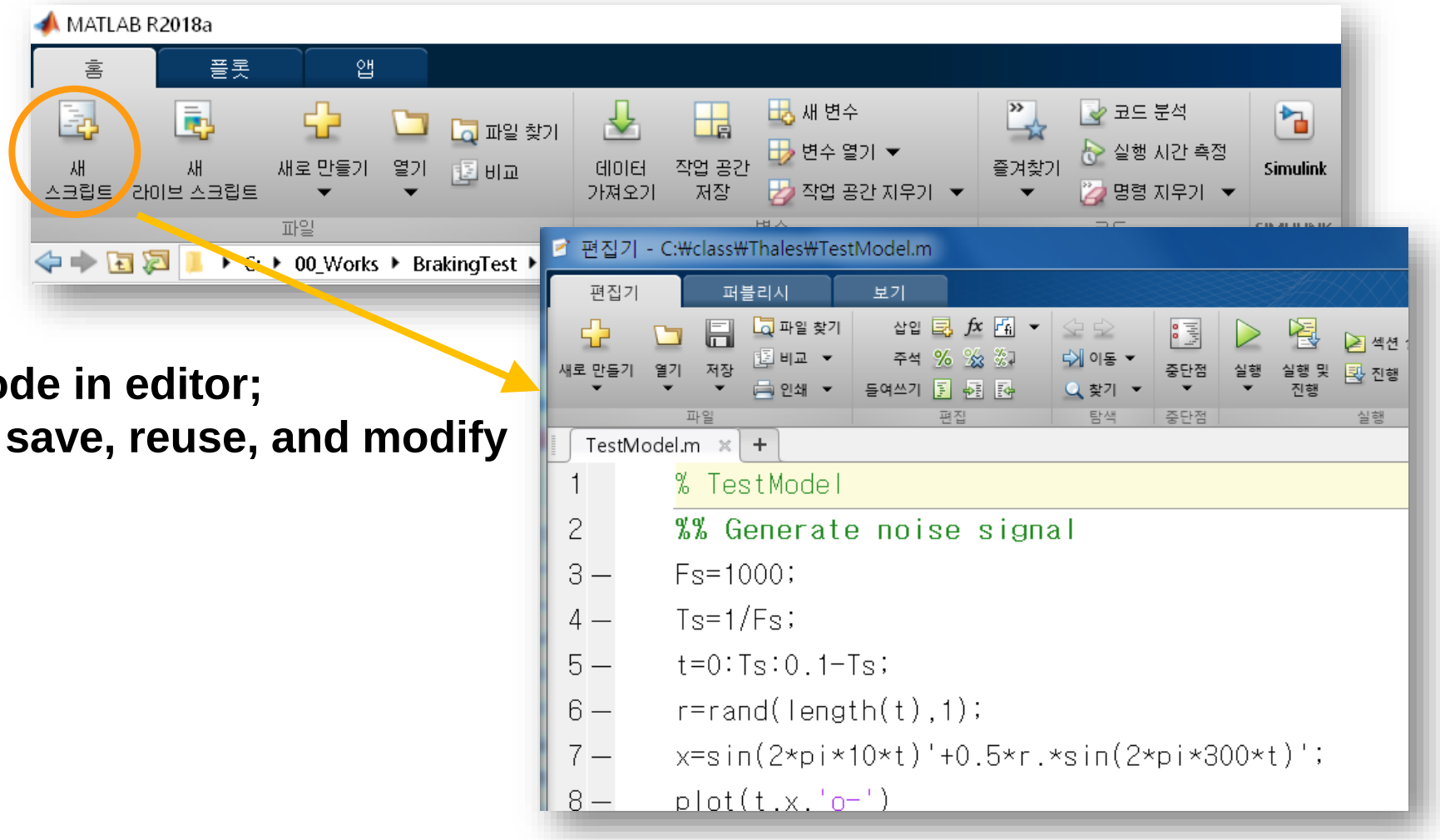
Programming Keywords

```
switch expression
case value1
   statement1
case value2
   statement2
otherwise
   statement3
end
```

```
>> a=0; b=0;
for i=1:10
switch i
case {1 3 5 7 9}
a=a+i;
case {2 4 6 8 10}
b=b+i;
otherwise
error('quit');
end
end
disp([a b])
```

Script

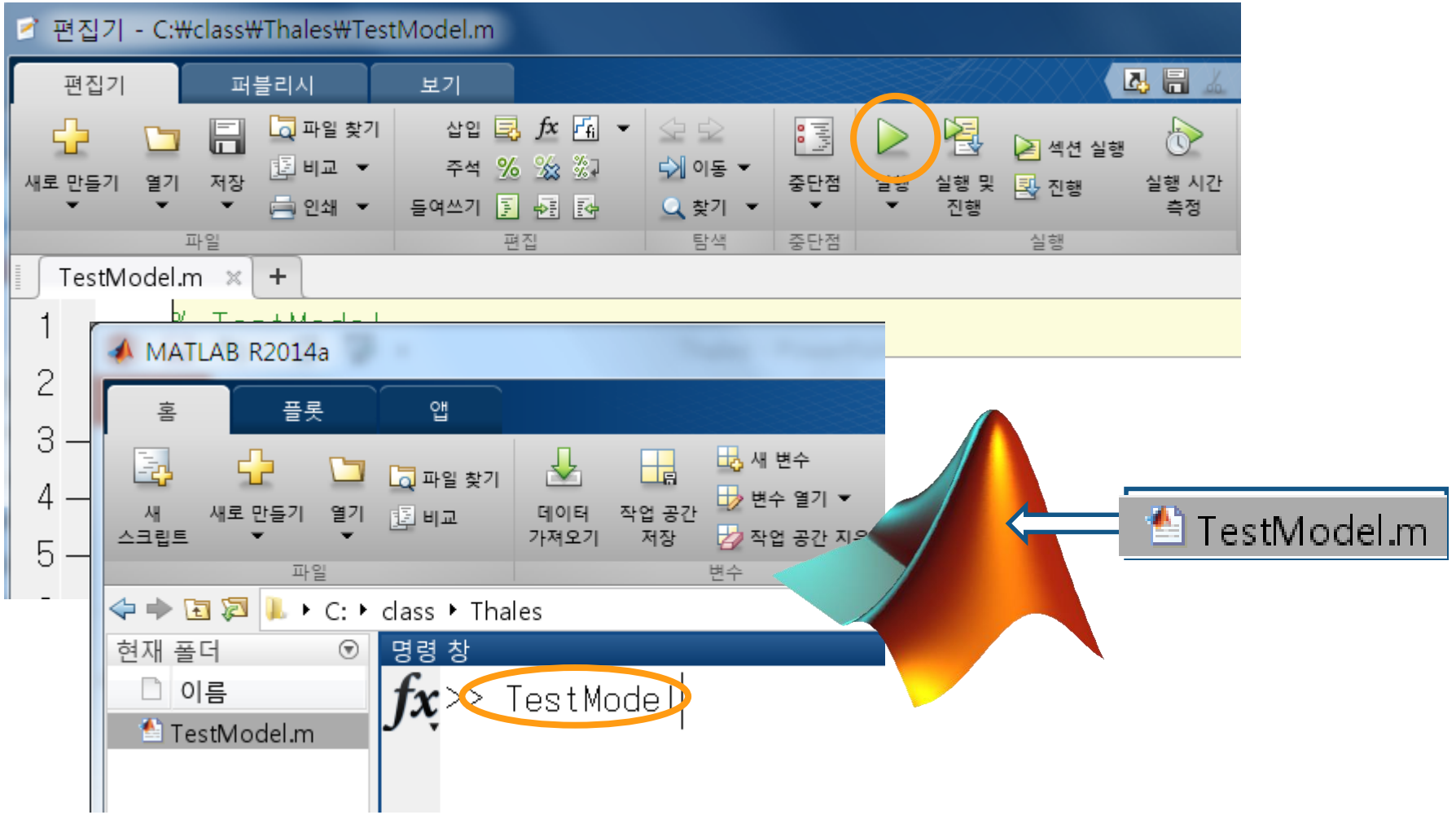
Script File



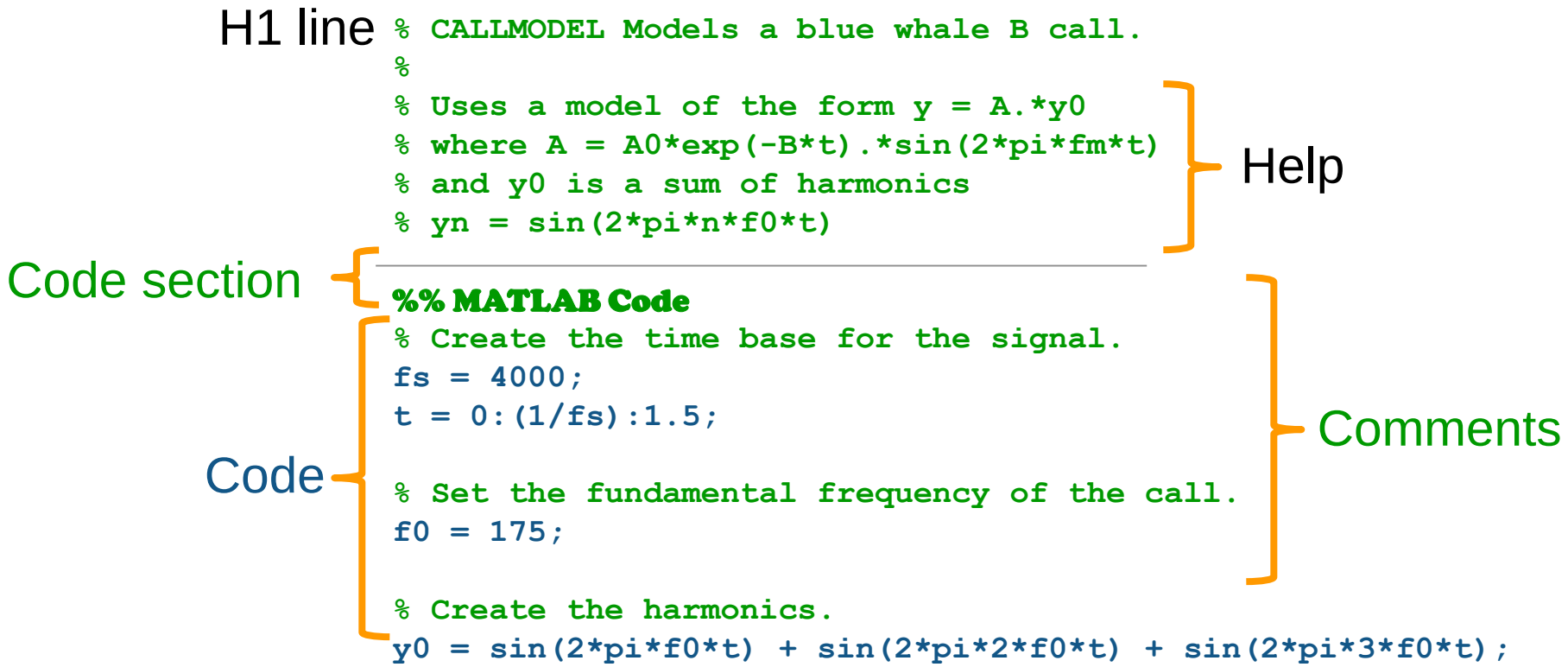
Write code in editor;
Easy to save, reuse, and modify

Running a Script

Press F5



Comment and Code Section

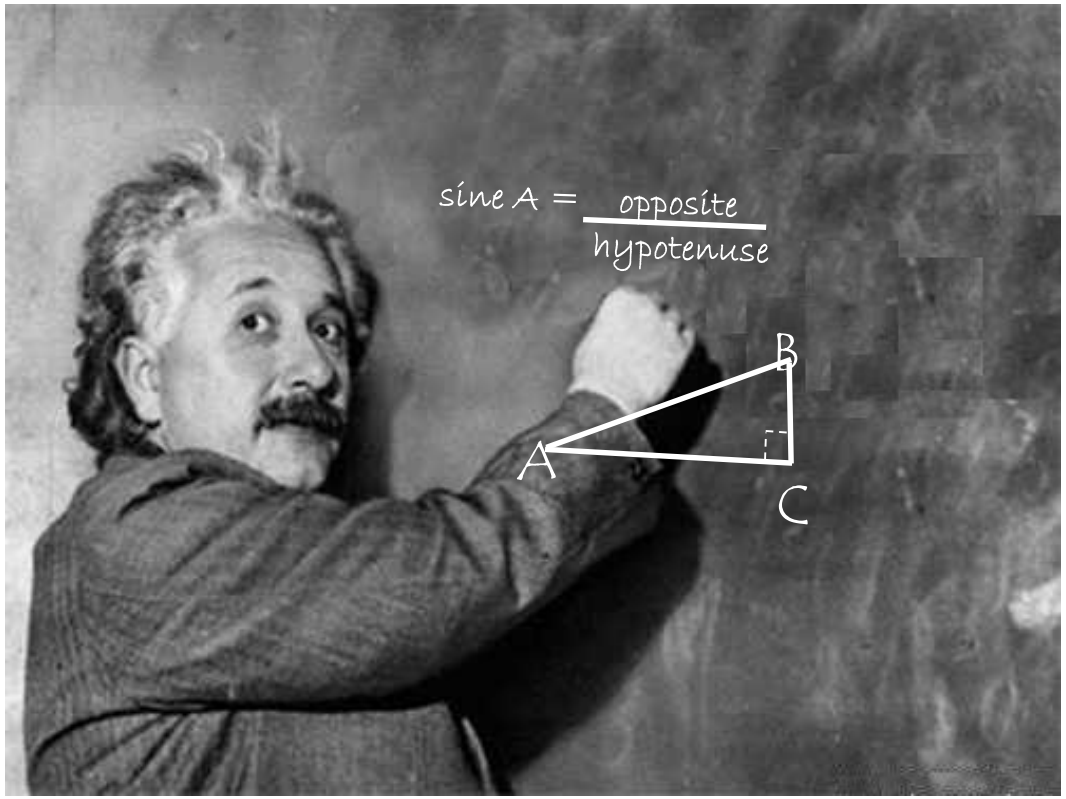


Function

What is a MATLAB® function?

- MATLAB provides elementary mathematical and statistical functions: abs, sqrt, exp, mean, std, sin, cos, etc.

- MATLAB also provides advanced mathematical functions:
  matrix inverse, matrix eigenvalues, Bessel and gamma functions, fast Fourier transforms, etc.



How to Create and Use Function

function [arg1, arg2, …]= func_name(arg1, arg2, …)

```
% TestModel
%% Generate noise signal
Fs=1000;
Ts=1/Fs;
t=0:Ts:0.1-Ts;
r=rand(length(t),1);
x=sin(2*pi*10*t)'+0.5*r.*sin(2*pi*300*t)';
plot(t,x,'o-')
m=20;
s=smoothing(x,m);
hold on; plot(t,s,'r*-')
%% FFT analysis
X=fft(x);
S=fft(s);
n=length(x);
f=(0:n-1)*(Fs/n);
z=conj(X).*X/n;
zz=conj(S).*S/n;
figure; stem(f,[z,zz])
```

x m

s

```
function s=smoothing(x,m)
w=ones(m,1)/m;
s=conv(x,w,'same');
```

Easy to reuse and maintain
Shorten programming code

>> arg1=func_name(arg1,arg2)

Create MATLAB function

```
function [p, m] = plusmin(a,b)
  p = a + b;
  m = a - b;
end
```

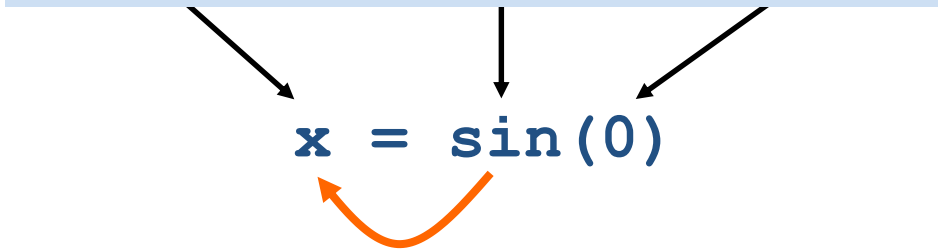
Save as plusmin.m

At Command Window

```
>>[a b] = plusmin(5,3)
  a =
     8
  b =
     2
```

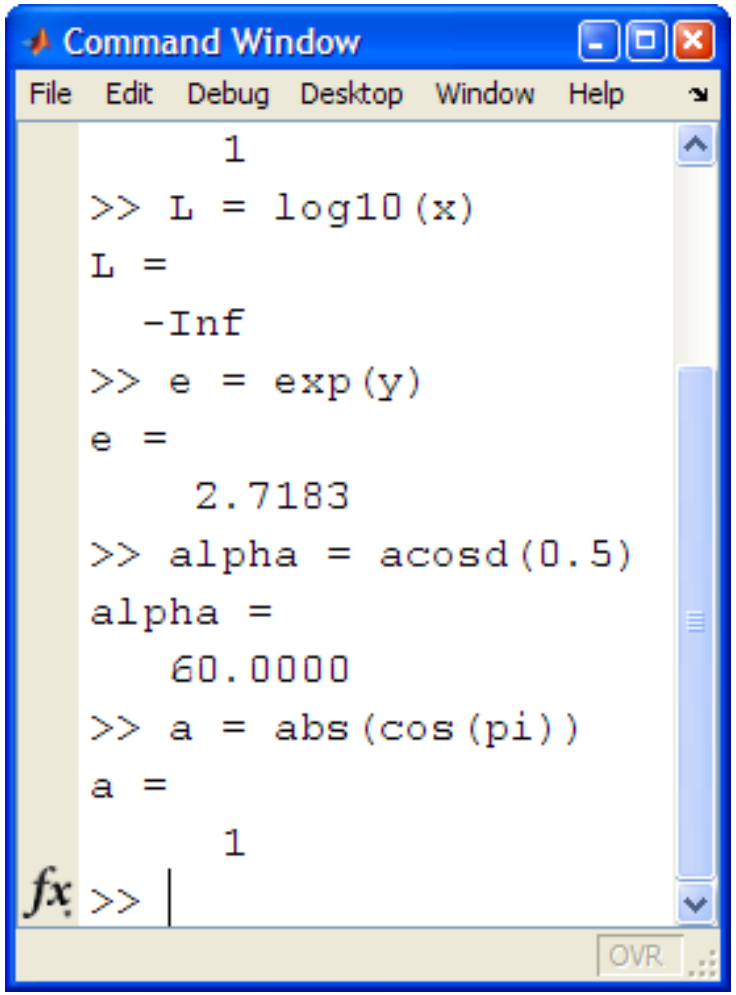
Basic Function Calling Syntax

```
output = functionName(input)
```



- The result of a function is assigned to the output variable.
- The input value is always entered in parenthesis after the function name.

How do I use basic MATLAB functions?

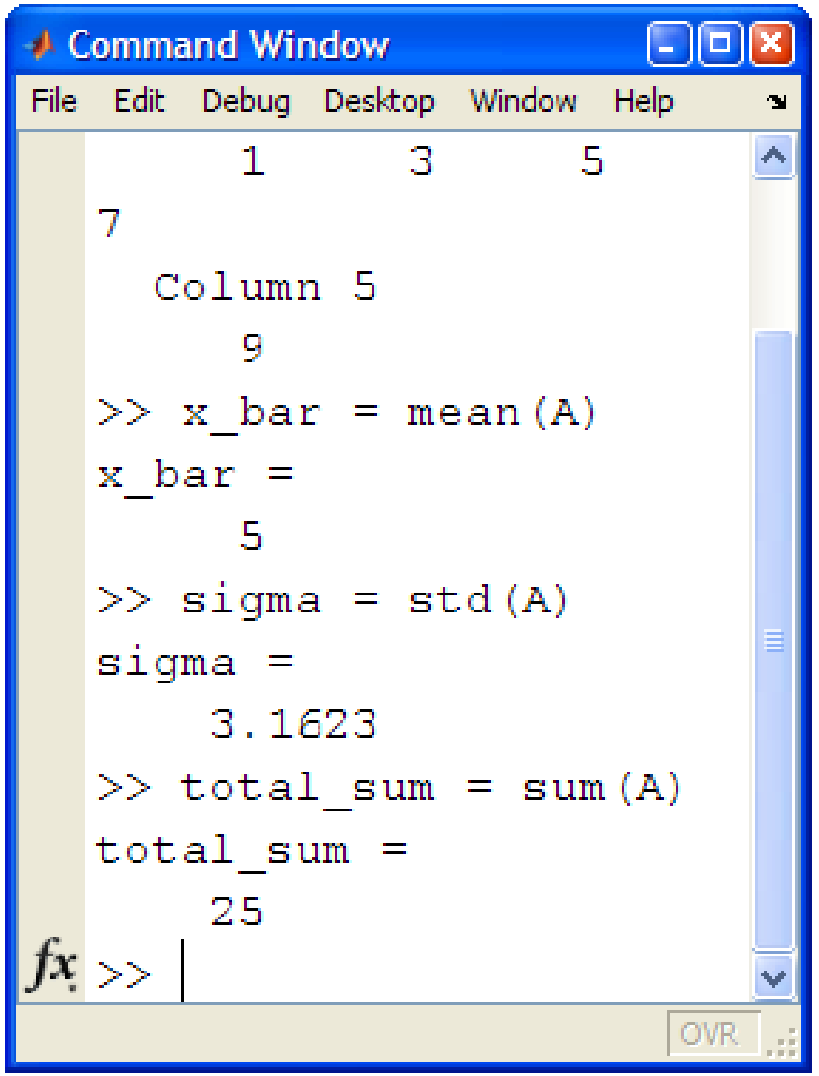


Basic Function Calling Syntax

```
output = functionName(input)
```

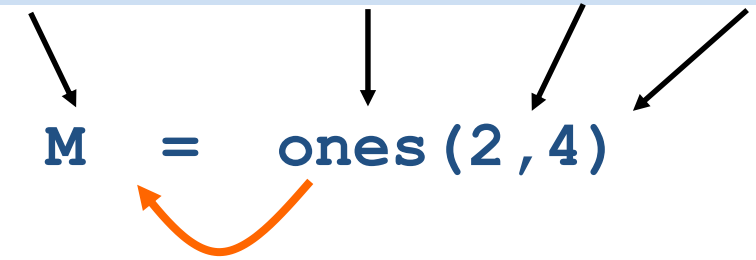
```
x_bar = mean(A)
```

Functions can operate on a vector or matrix.

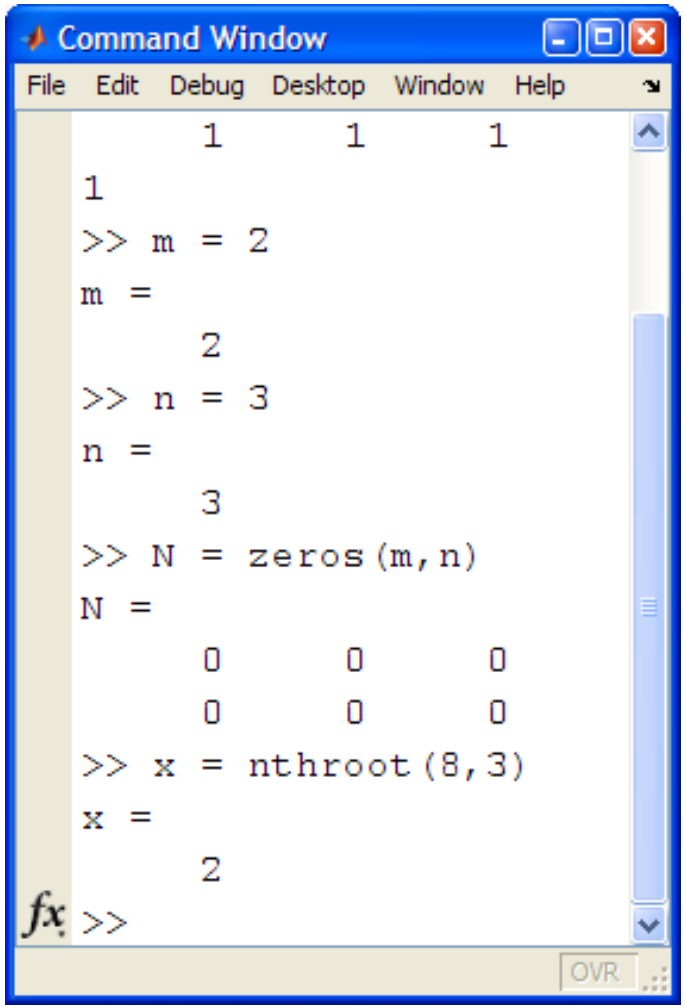


Function Calling Syntax

```
output = functionName(input1,input2)
```



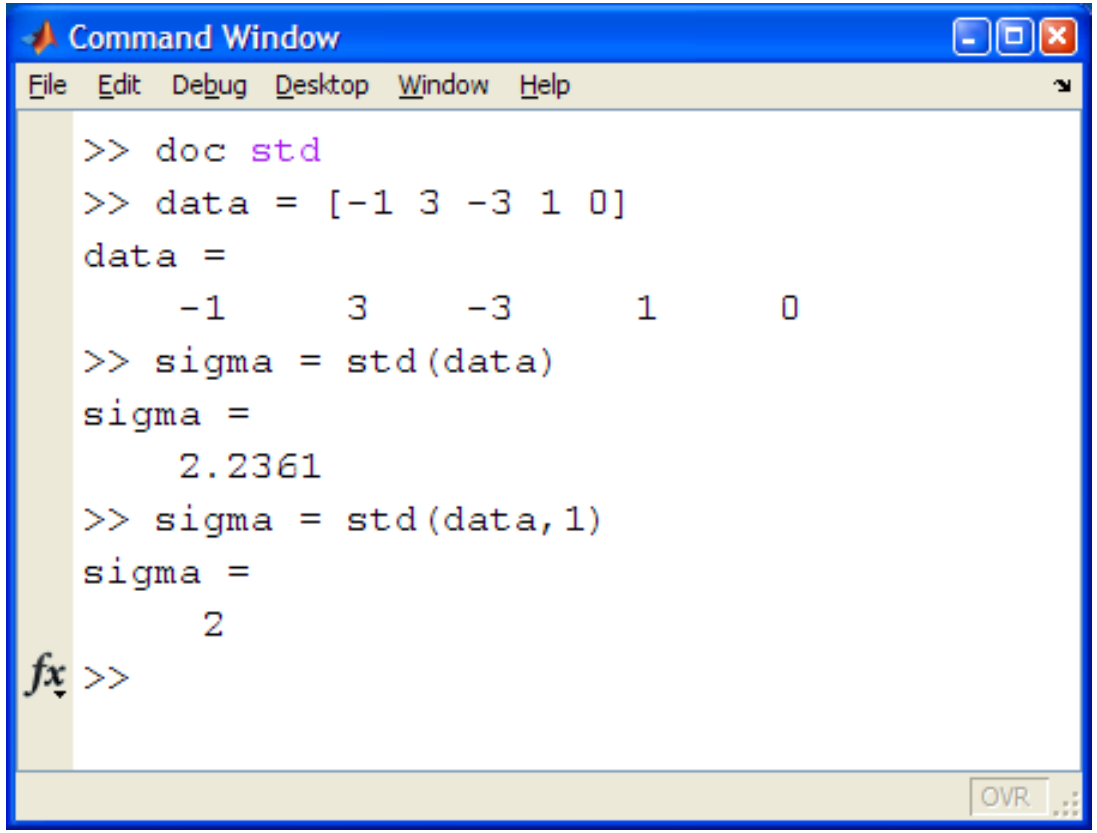
Functions can have multiple input variables



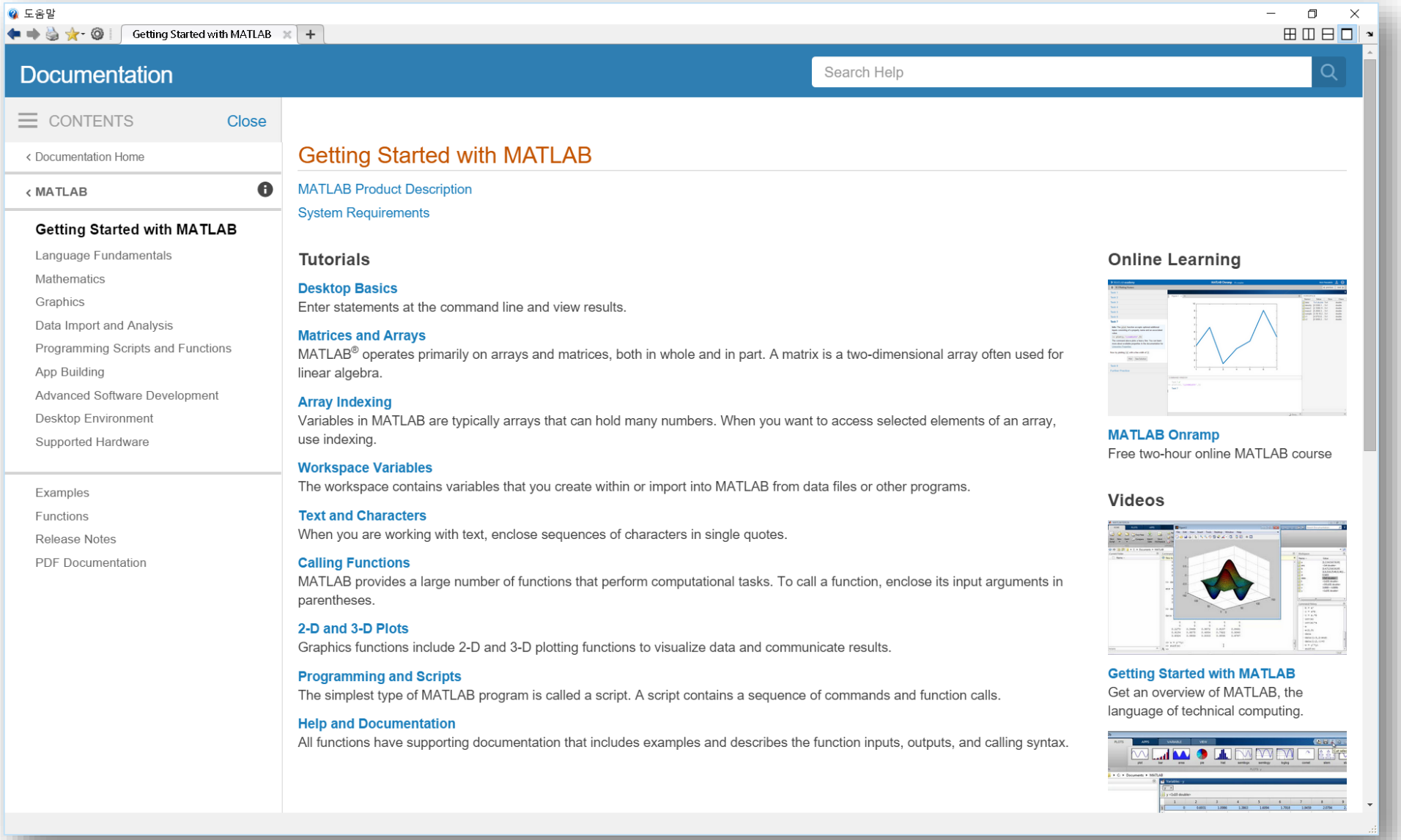
MATLAB Documentation

- The same function can be used in different ways.
- The documentation lists all the options for each function.

```
>> doc std
```



Getting Started with MATLAB



web(fullfile(docroot, 'matlab/getting-started-with-matlab.html'))

MATLAB Download & Installation

아카데미아

MathWorks.com내 검색

학생 대상 | 교육자 대상 | 연구원 대상

MATLAB 및 Simulink 교육과 학습

전 세계 5,000개 이상의 대학에서 사용하고 있습니다.

학생용 소프트웨어 받기 | 기본 배우기 | 강의 준비하기 | 프로젝트 시작하기

학생용 소프트웨어 받기

학생용 소프트웨어를 사용하여 학습 과정 및 연구를 성공적으로 이행하고, 취업 및 실무에 해당 지식을 활용하십시오.

MATLAB Student 구매

Looking for a campus-wide option?
Check out MATLAB for your campus

Which license is right for you?
Explore all license options

MATLAB Onramp

무료 인터랙티브 강의로 MATLAB을 경험하고 학습합니다.

자세히 보기

학생용 MATLAB

» MATLAB 구매

최우수 학생

» 학생들의 프로젝트 둘러보기

학생 대상 MATLAB 및 Simulink 챌린지

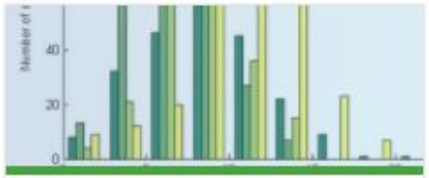
» 자세히 보기

학생

기술적인 영감을 주는 툴을 강의실은 물론, 업계 전반에서 활용할 수 있습니다. MATLAB 및 Simulink 기술과 함께하는 밝은 미래가 여러분을 기다립니다.

튜토리얼

» 양방향 수업 둘러보기



MATLAB 및 Simulink 예제

» 예제 보기

경진대회 및 콘테스트

» 소프트웨어 지원 및 도움말

Automotive

- American Solar Challenge
- BAJA SAEINDIA
- EcoCAR Mobility Challenge
- EducEco
- Formula SAE Brasil
- Formula SAE Japan
- Formula SAE® Michigan and Formula SAE® Lincoln
- Formula Student China
- Formula Student Germany
- Formula Student Spain
- Formula Student UK
- Range Extended Electric Vehicle (REEV)
- SAE AutoDrive Challenge
- SUPRA SAEINDIA

Robotics

- BEST Robotics
- Coupe de France de Robotique Challenge
- Droid Racing Challenge
- European Rover Challenge
- High School Autonomous Vehicle Challenge
- Intelligent Ground Vehicle Competition
- Micromouse Contest
- Pan-African Robotics Competition
- ROBOCON India
- ROBO-ONE
- RoboCup
  - RoboCupJunior
- RoboNation Competitions
- RoboRace
- Smart Model Car Contest
- Turtlebot3 AutoRace
- VEX Robotics
- World Robot Summit
- Zero Robotics

The Winner's Circle

See how students are winning competitions worldwide with MATLAB and Simulink

» Explore student projects

MATLAB and Simulink Student Challenges

» Learn more

지능형 모델 자동차 경진대회

지능형 모델 자동차 경진대회는 학생 팀들이 안티록 브레이크와 컴퓨터식 엔진 관리를 위한 안전 및 효율 시스템 개발에 사용되는 것과 동일한 기술인 ICT(Intelligent Car Technology)를 사용하여 모델 자동차를 설계, 제작하여 직접 경주해보는 대회입니다. 여러분의 팀은 MATLAB 및 Simulink를 활용한 모델 기반 설계를 적용함으로써 보다 효율적인 제어, 신호 처리, 이미지 및 비디오 처리, 메카트로닉스와 신뢰성 있는 임베디드 시스템으로 주어진 도전 과제를 수행해 낼 수 있습니다.

MathWorks는 지능형 모델 자동차 경진대회 참가 학생들에게 MATLAB 및 Simulink 소프트웨어를 제공합니다. 학생들이 MATLAB 및 Simulink를 모델 자동차 개발에 어떻게 활용하는지에 대해 보고서를 제출하면, MathWorks에서 타임랩(time-lap)과 보고서 점수를 기반으로 하여 수상자 세 팀을 선정합니다.

무료 소프트웨어

MathWorks에서는 이번 대회를 위해 무료 소프트웨어를 제공합니다. 대회에 참가하는 팀이고 소프트웨어가 필요한 경우, 소프트웨어 신청 양식을 작성하십시오.

소프트웨어 신청

MATLAB 및 Simulink 레이싱 라운지

계속해서 개발되는 이 비디오 시리즈를 통해 MATLAB 및 Simulink의 기본 사항과 자동차 개발에 대한 자세한 정보를 확인하십시오.

비디오 보기

학생경진대회 소프트웨어 신청방법

MathWorks | Offer of Complimentary Software License(s)

MathWorks is pleased to sponsor Smart Car. MathWorks will provide software, training, and access to MathWorks engineering mentors and technical support to teams who have completed the Student Competition Software Request Form.

Software Request Form

Here are the products we will be providing that you may have access to:

Complimentary Software for

- MATLAB
- Simulink
- Control System Toolbox
- DSP System Toolbox
- Embedded Coder
- Filter Design HDL Coder
- Fixed-Point Designer
- Global Optimization Toolbox
- HDL Coder
- MATLAB Coder
- MATLAB Report Generator
- Neural Network Toolbox
- Optimization Toolbox
- Signal Processing Toolbox
- Simscape Driveline
- Simscape Electronics
- Simscape Multibody
- Simscape Power Systems
- Simscape
- Simulink Coder
- Simulink Control Design
- Simulink Design Optimization
- Simulink Design Verifier
- Simulink Report Generator
- Simulink PLC Coder
- Simulink Verification and Validation
- Stateflow
- System Identification Toolbox

Type: Designated Computer
License option: Individual
Platform: All

Your team leader or faculty advisor should review and complete the Student Competition Software Request Form to take advantage of our software offer.

Software Request Form

Best of luck in your preparation!

Lauren Tabolinsky
MathWorks Student Competition Program

학생경진대회 소프트웨어 신청방법

MathWorks | Accelerating the pace of engineering and science

한국 | 연락처 | 구입 방법 Search MathWorks

Hwang Eunju | My account | 로그아웃

제품 솔루션 대학 커리큘럼 지원 User Community 이벤트 회사소개

Academia

피드백

학생 경진 대회 소프트웨어 신청양식

신청을 희망하는 팀의 리더 혹은 학과 담당자가 아래의 양식을 기입하여 주시기 바랍니다.

"How will you use MathWorks software?"라는 질문에는 "Academic Use"를 선택하여 주십시오. 본 신청양식을 기입함으로써 학생경진대회를 위한 소프트웨어 신청이 이루어지게 됩니다. 신청에 대한 답변은 매스웍스에서 business hour 72시간 안에 답변 드릴 것입니다. 본 신청이 승인되면, 소프트웨어 다운로드를 위한 가이드 및 안내를 받게 됩니다.

만약에 매스웍스의 계정이 없으시다면, 지금 바로 매스웍스 홈페이지 (www.mathworks.com)에서계정을 생성하여 주십시오. 본 신청은 귀하의 이메일 주소를 통해 가입 확인 절차가 이루어지게 됩니다.

* Indicates required information

안녕하십니까 Eunju Hwang

Team Leader or Faculty Advisor

*Team Leader/Faculty Advisor Confirmation

팀 리더/학과 담당자임을 확인합니다. (아래 체크하여 주십시오)

신청자 영문 정보

*회사/학교명 (영문)

회사 혹은 학교의 공식 명칭을 입력해 주십시오.

Competition Information - Complete in English

*참가를 희망하는 경진대회 이름 (ex. Smart Car Contest)

*대학교 이름

*팀 이름

*총 팀원수 – 매스웍스 소프트웨어를 사용하게 될 인원 수

MATLAB R2019a 설치 및 실행