



Oprogramowanie
Naukowo-Techniczne
sp. z o.o.

Kraków, 19.10.2023 r.

RoadRunner Introduction

3D Scenes and Scenarios for ADAS Design and Verification

Mateusz Łabęcki

Application Engineer ONT

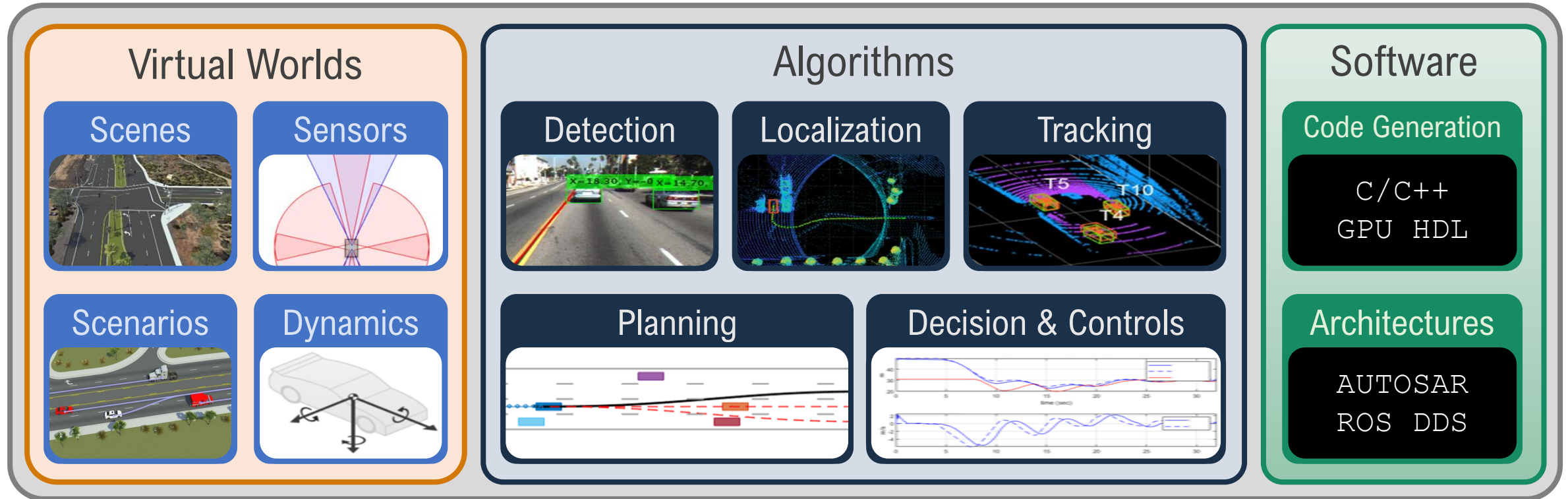


Agenda

1. Introduction
2. RoadRunner Product Family
3. ADAS Simulation Example

Introduction

Automated Driving Applications Development



RoadRunner
product family

MATLAB & Simulink
product family

Polyspace
product family

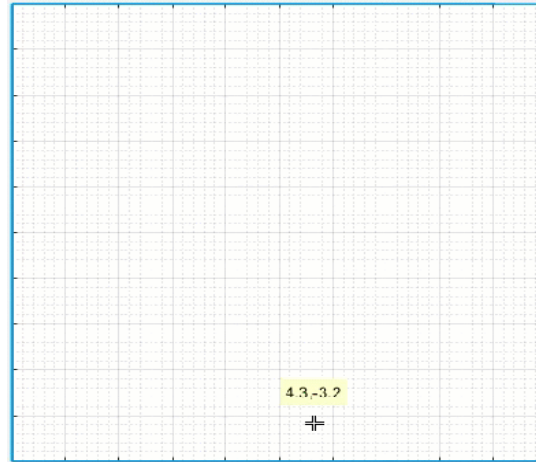
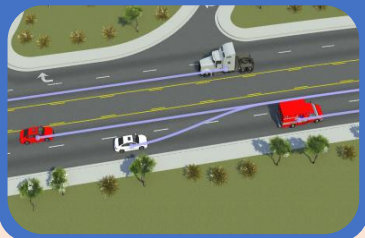
Complex Scenes and Scenarios for Today's Needs

Virtual Worlds

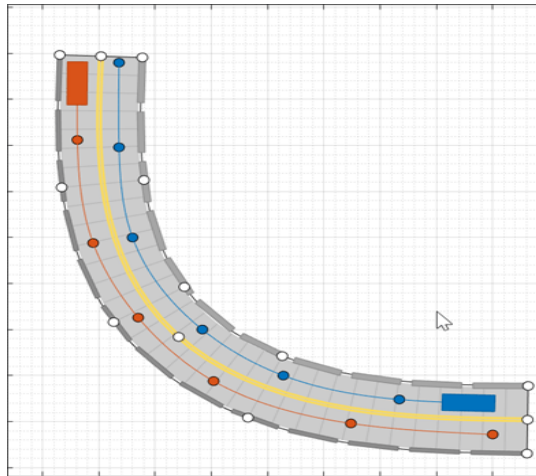
Scenes



Scenarios



OR



OR



Complexity and realism

Accurate Road Models



Real Photo – Shanghai, China



RoadRunner

Continuous investment in virtual simulation for design & verification

Verify & Validate

Analyze
Recorded
Data

Design
Virtual
Worlds

Design
Algorithms &
Systems

Design
Software

Integrate with External Tools and Software

Representative scenes and scenarios to maximize simulation value



Increased automation of building scenes and scenarios for recorded data



Interactive tools to design new and refine automatically built world

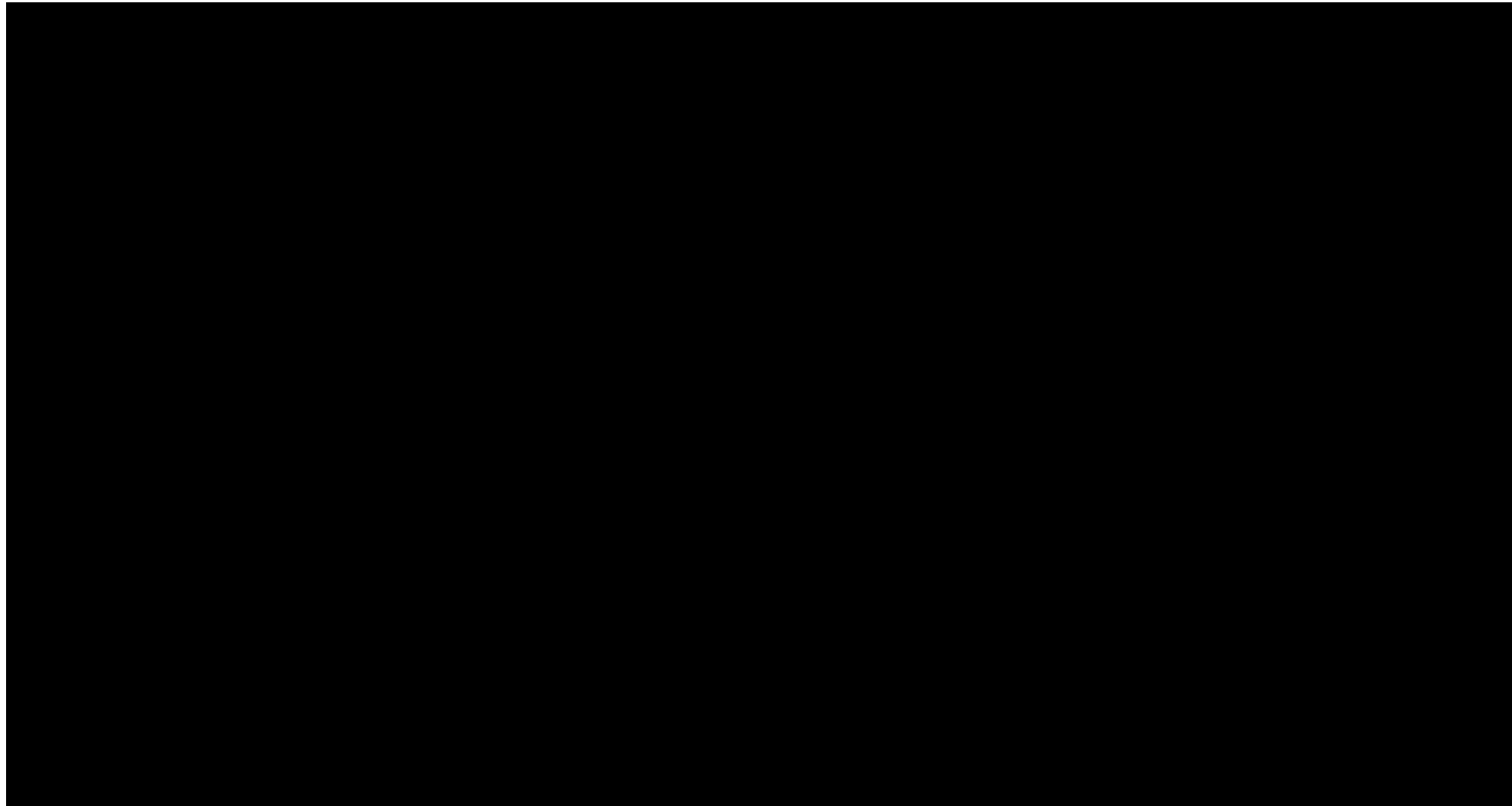


Tool interoperability to work with multiple driving simulation environments

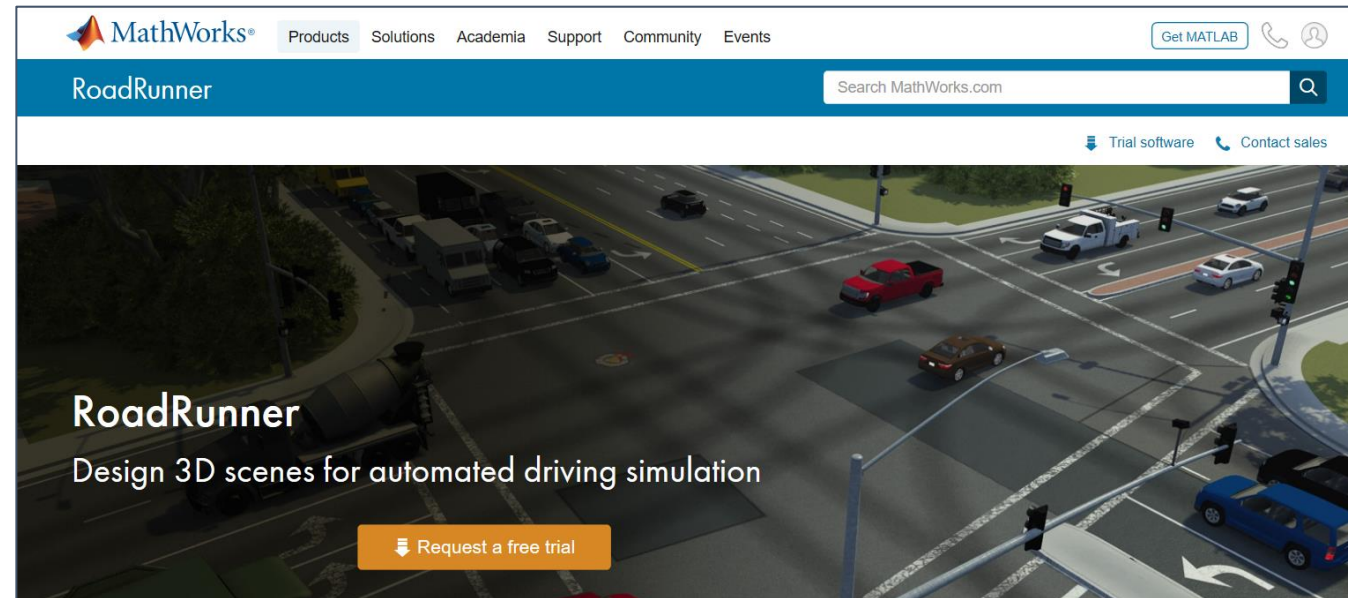


RoadRunner Product Family

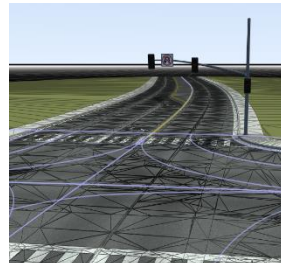
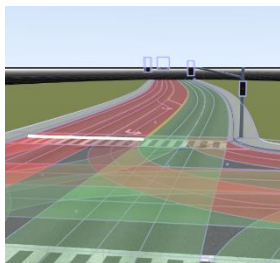
RoadRunner Video Presentation



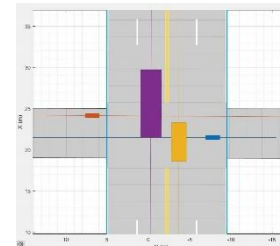
Use of RoadRunner Environment



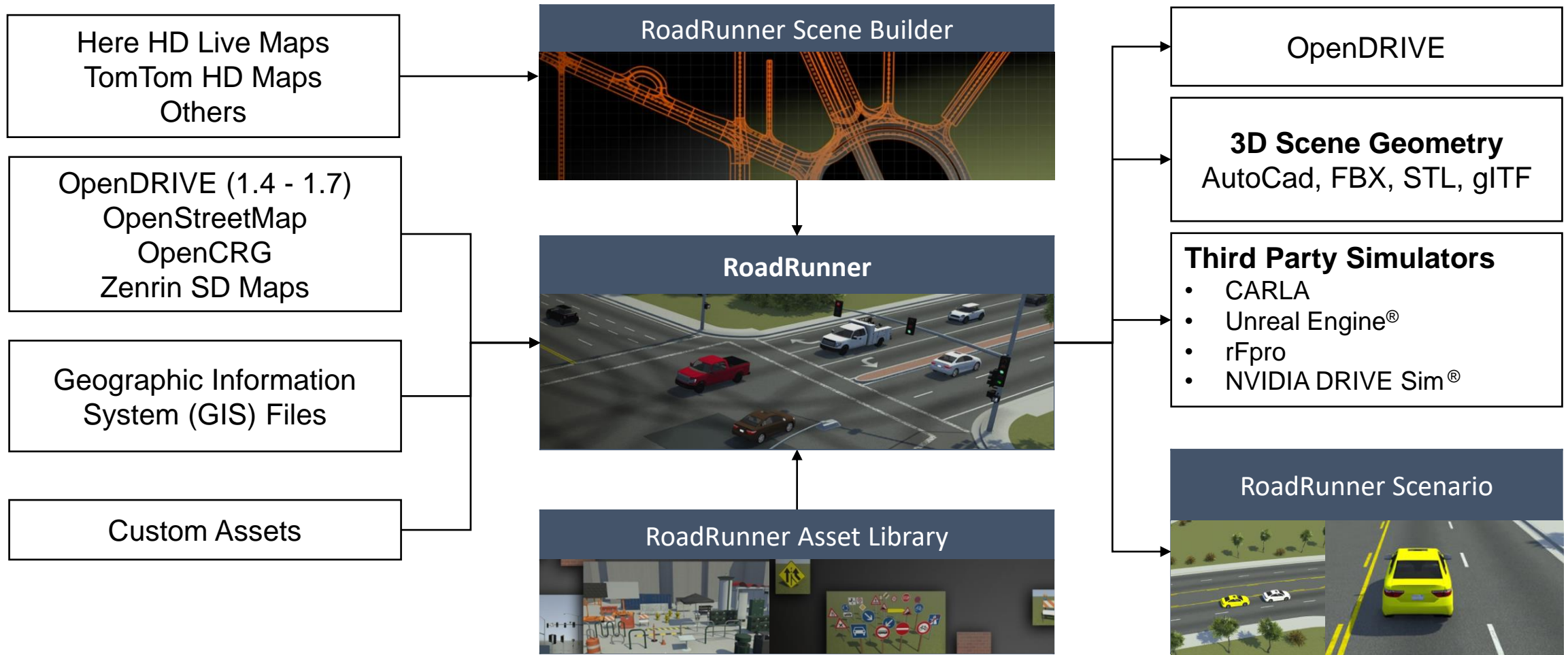
External Simulators



MATLAB & Simulink

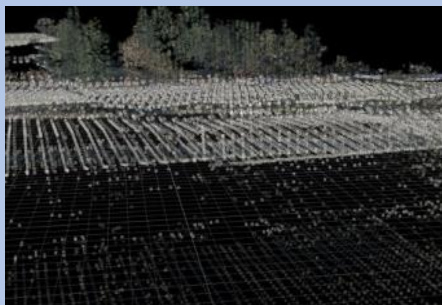
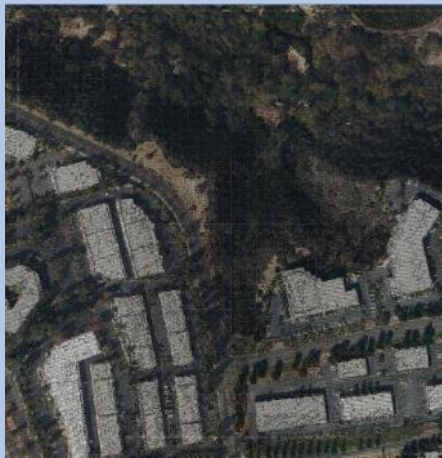


RoadRunner Product Family

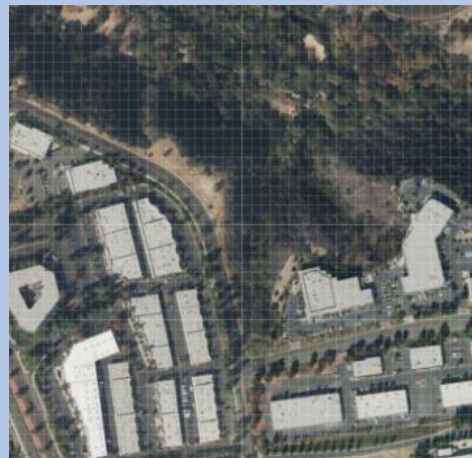


GIS File Interpretation

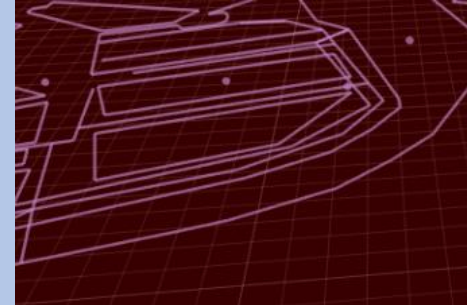
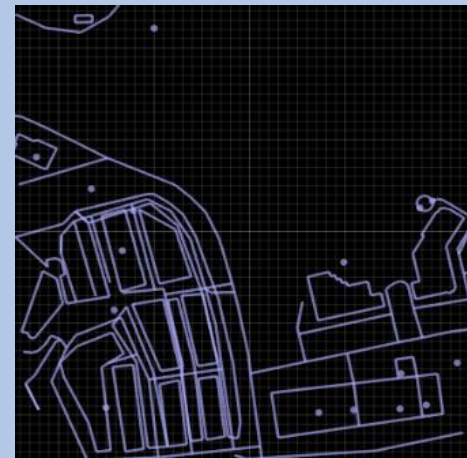
Point Cloud



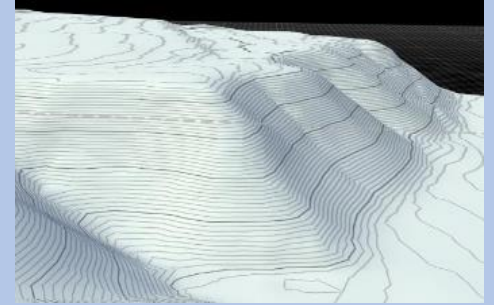
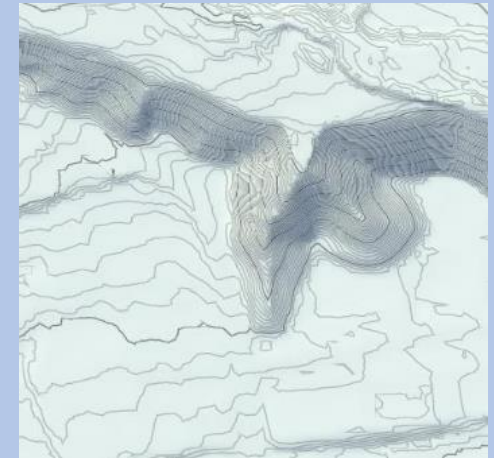
Aerial Imagery



Vector Data

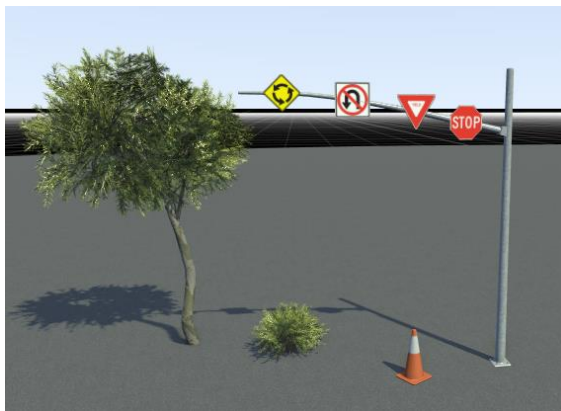


Elevation Data

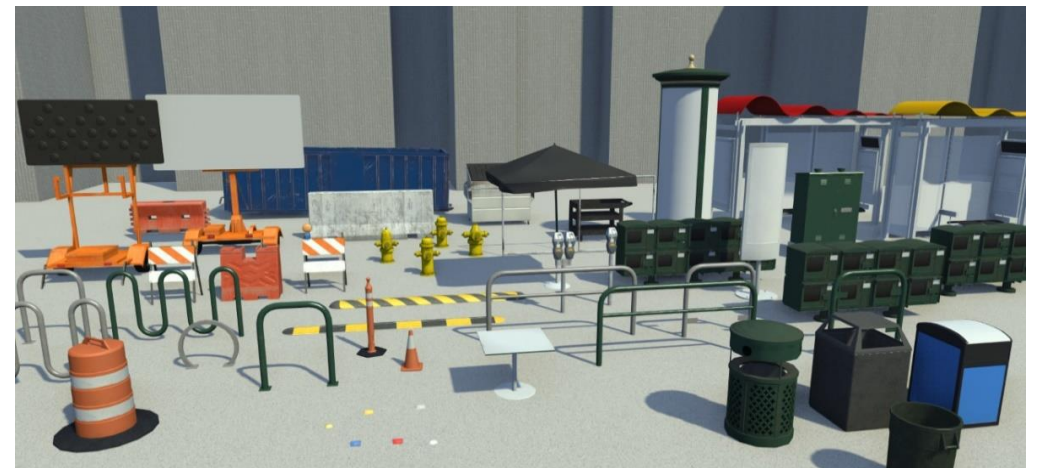


RoadRunner Asset Library

- Road signs
- Traffic signals
- Road surface markings and textures
- Buildings
- Guardrails and urban items
- Trees

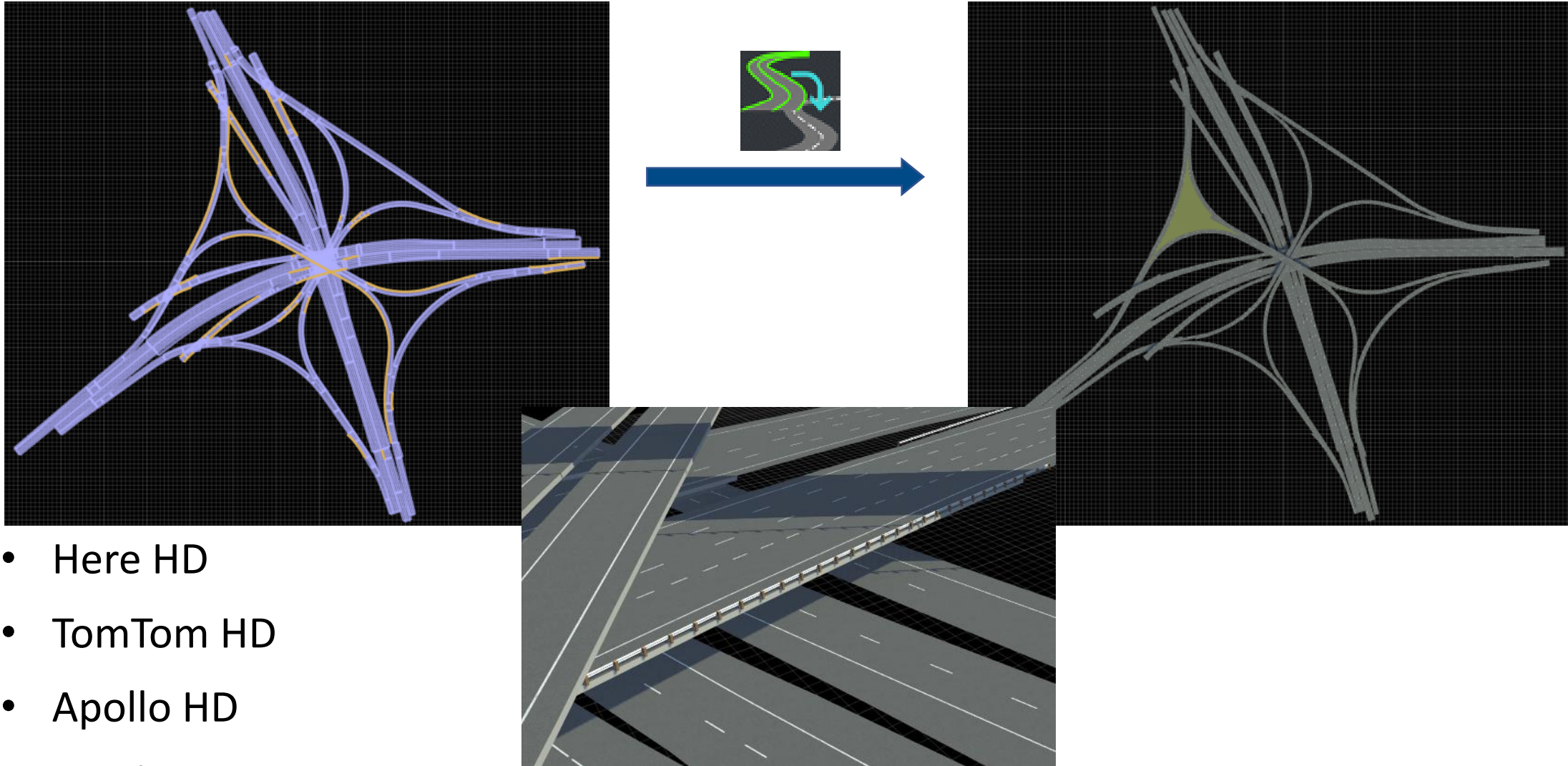


Basic models



Asset Library Models

RoadRunner Scene Builder Tool




- Here HD
- TomTom HD
- Apollo HD
- Zenrin Japan Map

RoadRunner Scenario

SpeedBump Actions.rscenario | 22a Project | MathWorks RoadRunner R2022a

File Edit View Tools Assets Window Help

Scenario Editing



Simulation

Simulation Controls

Time: 1.640 s

Enable Pacing to Slow Down Simulation

Slower Faster

0.05x 1x 20x

Simulation Properties

Step Size: 0.02000 s Max Time: 1000.000

Camera

Camera View: Follow

Actor: Car

Distance: 5.000

Height: 3.000

2D Editor | Logic Playback

Variables

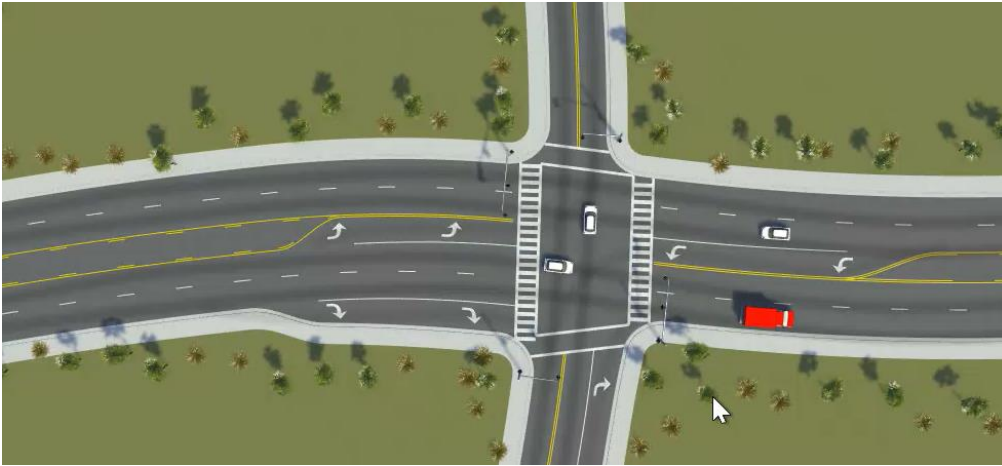
Name	
1 Hatchback_InitialSpeed	14
2 Car_NumLanesToChange	2
3 Car_LaneChangeDirection	LeftOf
4 Car_DistanceBehindSpeedBump	-17.98385

Simulation Tool

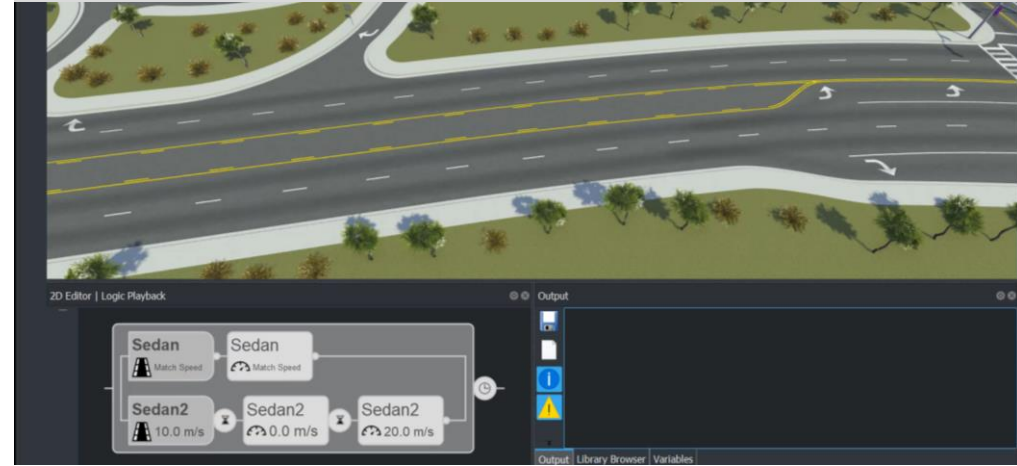
MathWorks

Scenario Logic and Paths

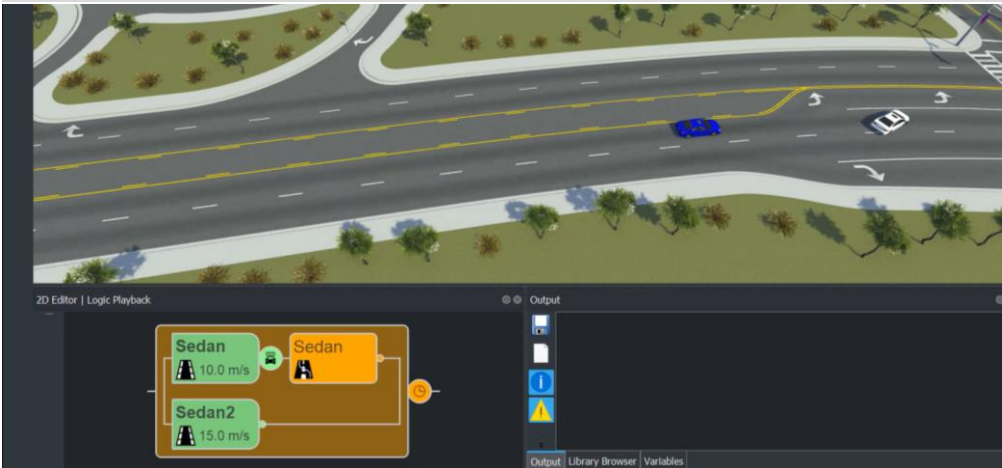
Follow lanes when no path is specified



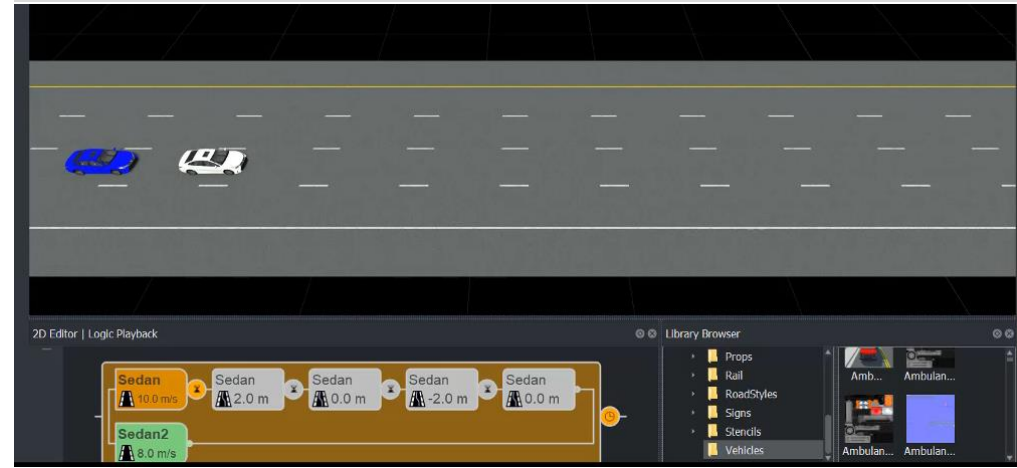
Speed actions



Lane change actions



Lateral offset actions



Programmatic Scene and Scenario Management

RoadRunner API

- Starting/Closing RoadRunner
- Change scenario variables
- Running scenario simulations
- Data acquisition such as trajectory and speed
- Export to ASAM formats
- Save/Create/Open
- Plot



MATLAB
 Automated Driving
 Toolbox

Python

C++

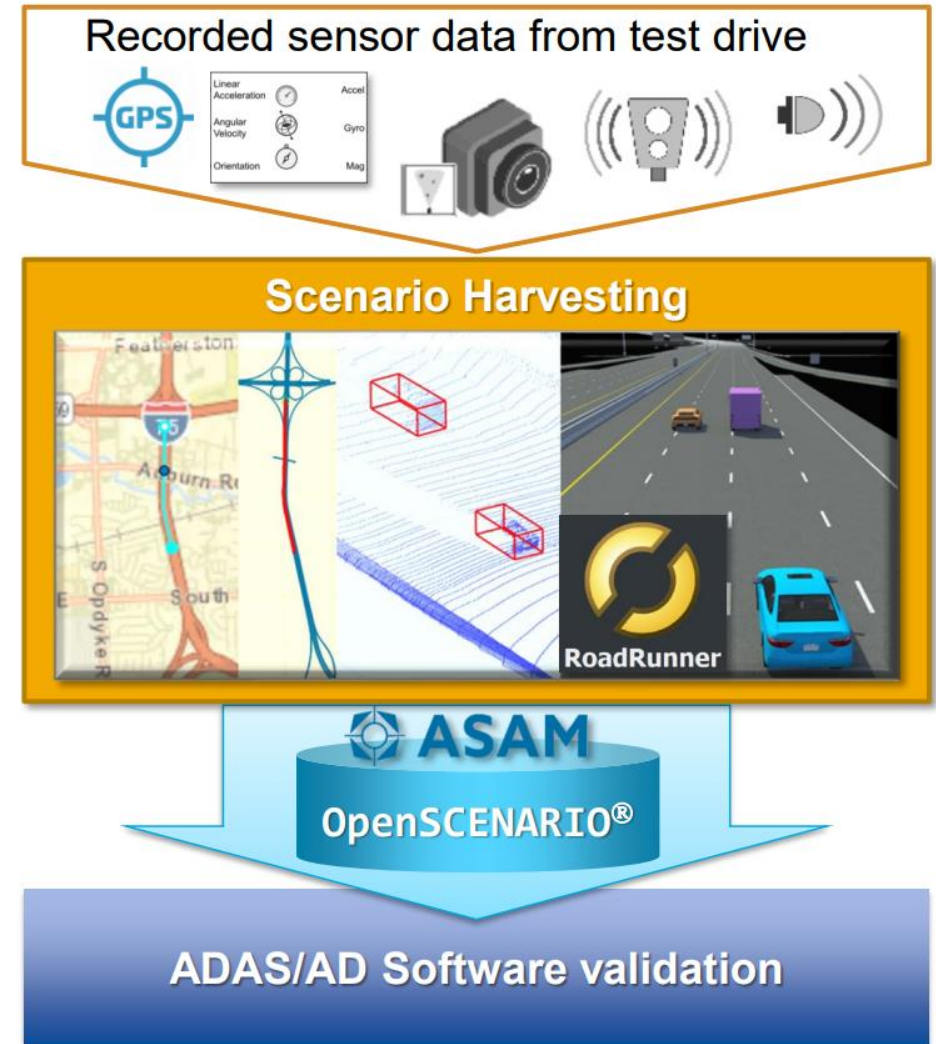
gRPC

Digital Twin of real-world scenarios

MathWorks
**AUTOMOTIVE
CONFERENCE 2023**
North America

**Scenario Harvesting from Recorded
Sensor Data using Automated
Driving Toolbox and Roadrunner
Scenario**

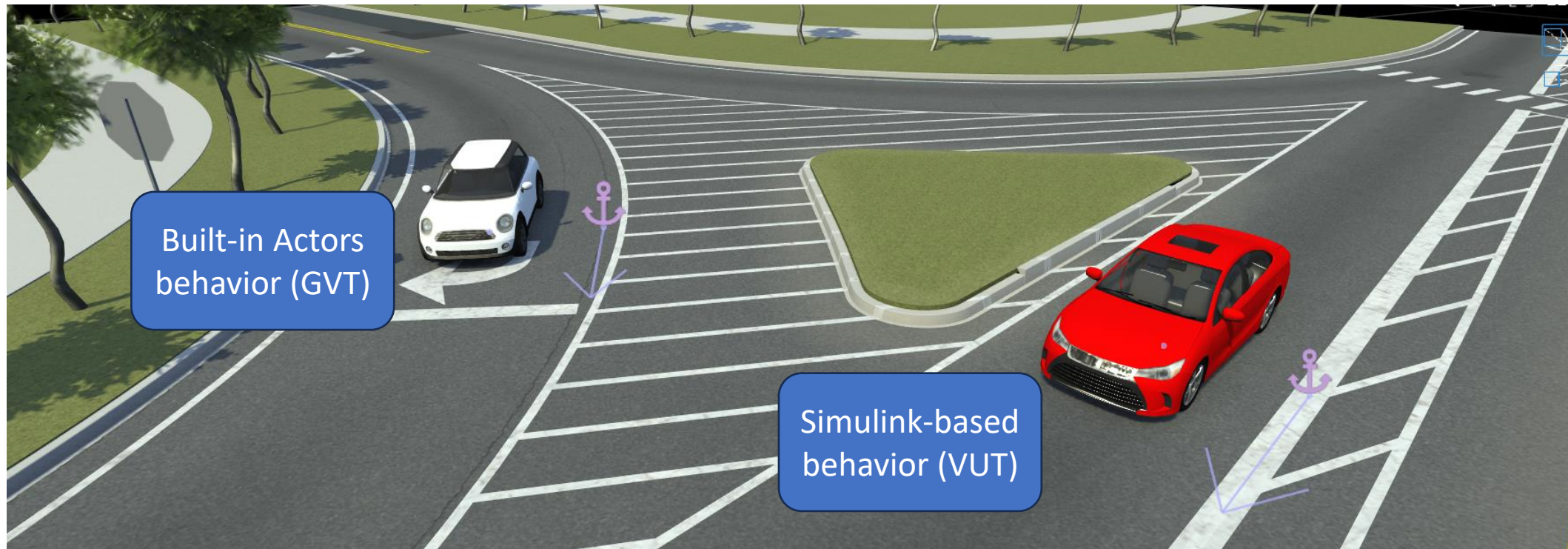
Krishna Koravadi, Aptiv Seo-Wook Park, MathWorks



ADAS Simulation Example

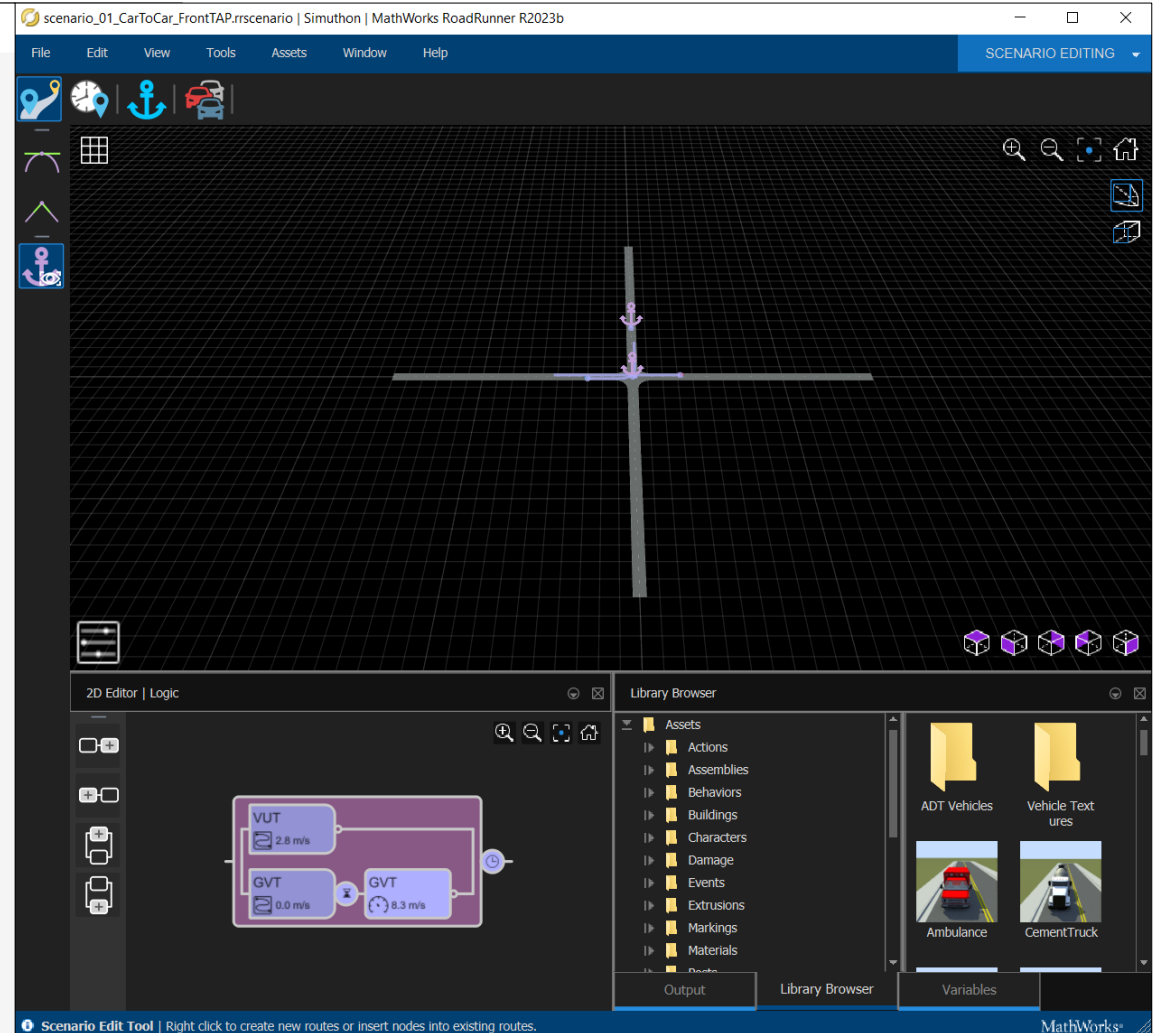
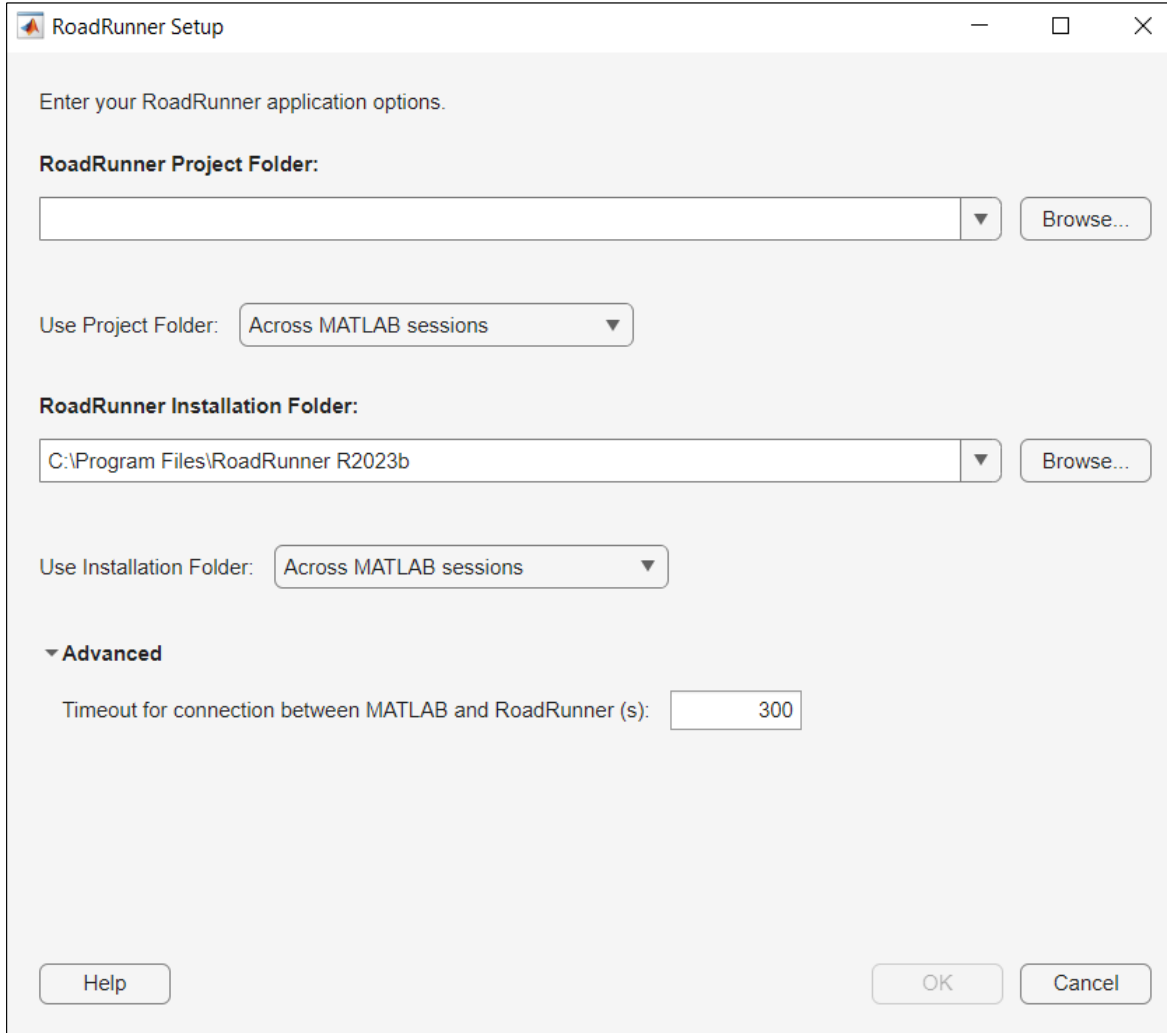
Connection Between MATLAB and RoadRunner

„Ok, I’ve got the scene and the scenario, now what?”



- **Actors can read:** actions, pose, velocity, actors dimensions, lanes and boundaries
- **Actors can write:** pose and velocity for each scenario simulation step

Step 1: Connect MATLAB to RoadRunner



```
>> rrApp = roadrunnerSetup
```

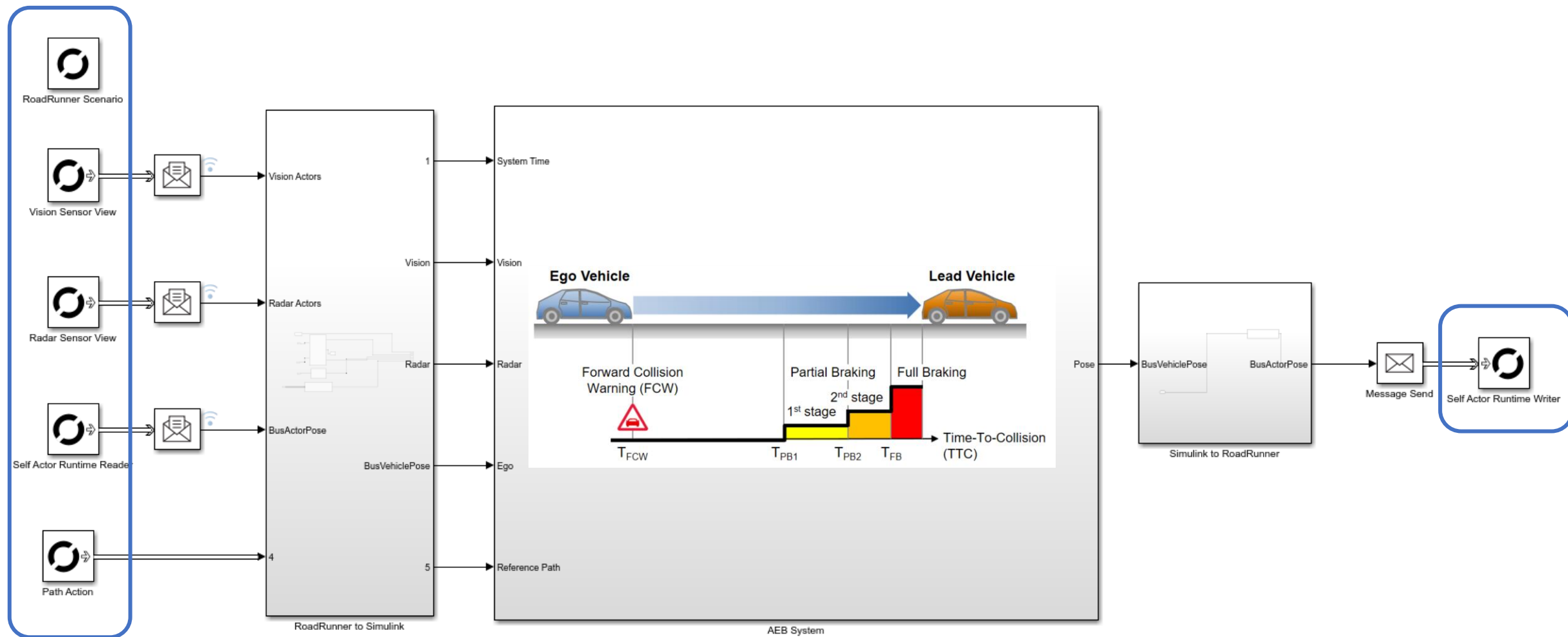
Step 2: Define VUT Behavior – RoadRunner

The screenshot displays the MathWorks RoadRunner software interface. The main window is titled "scenario_01_CarToCar_FrontTAP.rscenario | Simulink | MathWorks RoadRunner R2023b". The interface is divided into several sections:

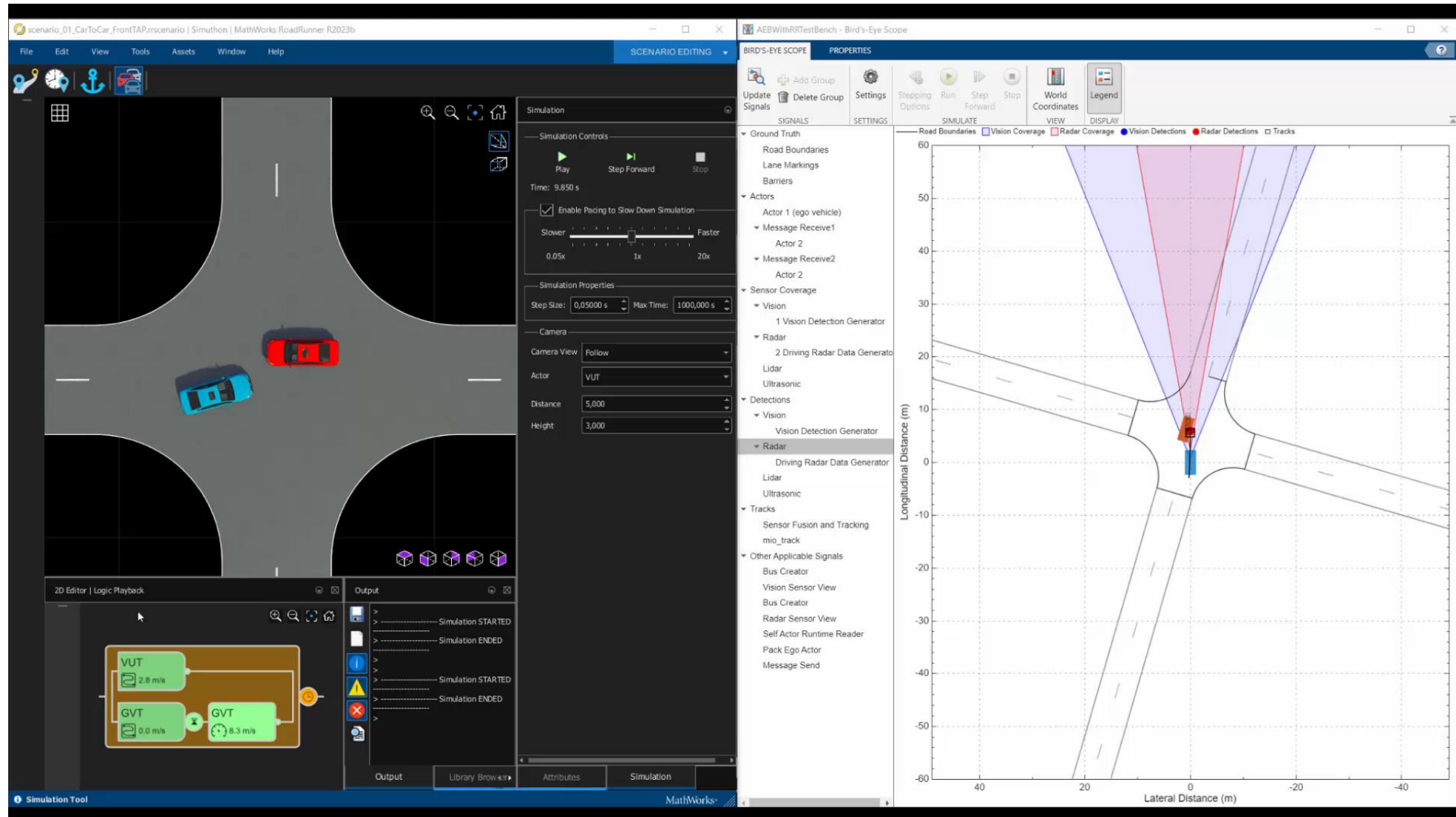
- Top Bar:** Contains menu options (File, Edit, View, Tools, Assets, Window, Help) and a "SCENARIO EDITING" dropdown.
- 2D Editor | Logic:** The central workspace shows a 2D view of a road intersection. A blue car is positioned on the left side of the road, and a red car is on the right. A purple line indicates a path or route. The interface includes various tool icons on the left and right sides.
- Logic Editor:** Located at the bottom left, it shows a logic diagram with three blocks: "VUT" (set to 2.8 m/s), "GVT" (set to 0.0 m/s), and another "GVT" (set to 8.3 m/s). The blocks are connected in a sequence.
- Library Browser:** Located at the bottom right, it displays a list of assets and folders, including Actions, Assemblies, Behaviors, Buildings, Characters, Damage, Events, Extrusions, Markings, and Materials.

At the bottom of the interface, there is a status bar with the text: "Scenario Edit Tool | Right click to create new routes or insert nodes into existing routes." The MathWorks logo is visible in the bottom right corner.

Step 2: Define VUT Behavior – MATLAB & Simulink



Step 3: Run Simulation



The screenshot displays the MATLAB/Simulink RoadRunner simulation environment. The main window is titled "scenario_01_CarToCar_FrontIAP.scenario | Simulink | MathWorks RoadRunner R2023b".

Simulation Controls:

- Buttons: Play, Step Forward, Stop
- Time: 9.850 s
- Enable Pacing to Slow Down Simulation:
- Speed slider: Slower (0.05x) to Faster (20x), currently at 1x
- Simulation Properties: Step Size: 0,05000 s, Max Time: 1000,000 s
- Camera: Camera View: Follow, Actor: VUT, Distance: 5,000, Height: 3,000

2D Editor | Logic Playback:

- Logic blocks: VUT (2.8 m/s), GVT (0.0 m/s), GVT (8.3 m/s)
- Output log: Simulation STARTED, Simulation ENDED

BIRD'S-EYE SCOPE:

- Legend: Road Boundaries, Lane Markings, Barriers, Vision Coverage, Radar Coverage, Vision Detections, Radar Detections, Tracks
- Actors: Actor 1 (ego vehicle), Actor 2
- Sensor Coverage: Vision (1 Vision Detection Generator), Radar (2 Driving Radar Data Generator)
- Tracks: Sensor Fusion and Tracking, mio_track
- Other Applicable Signals: Bus Creator, Vision Sensor View, Radar Sensor View, Self Actor Runtime Reader, Pack Ego Actor, Message Send

Bird's-Eye Scope Plot:

- Y-axis: Longitudinal Distance (m), ranging from -60 to 60
- X-axis: Lateral Distance (m), ranging from -40 to 40
- Visuals: Road boundaries, lane markings, and sensor coverage cones (Vision and Radar) centered on the ego vehicle (orange car) at (0,0).

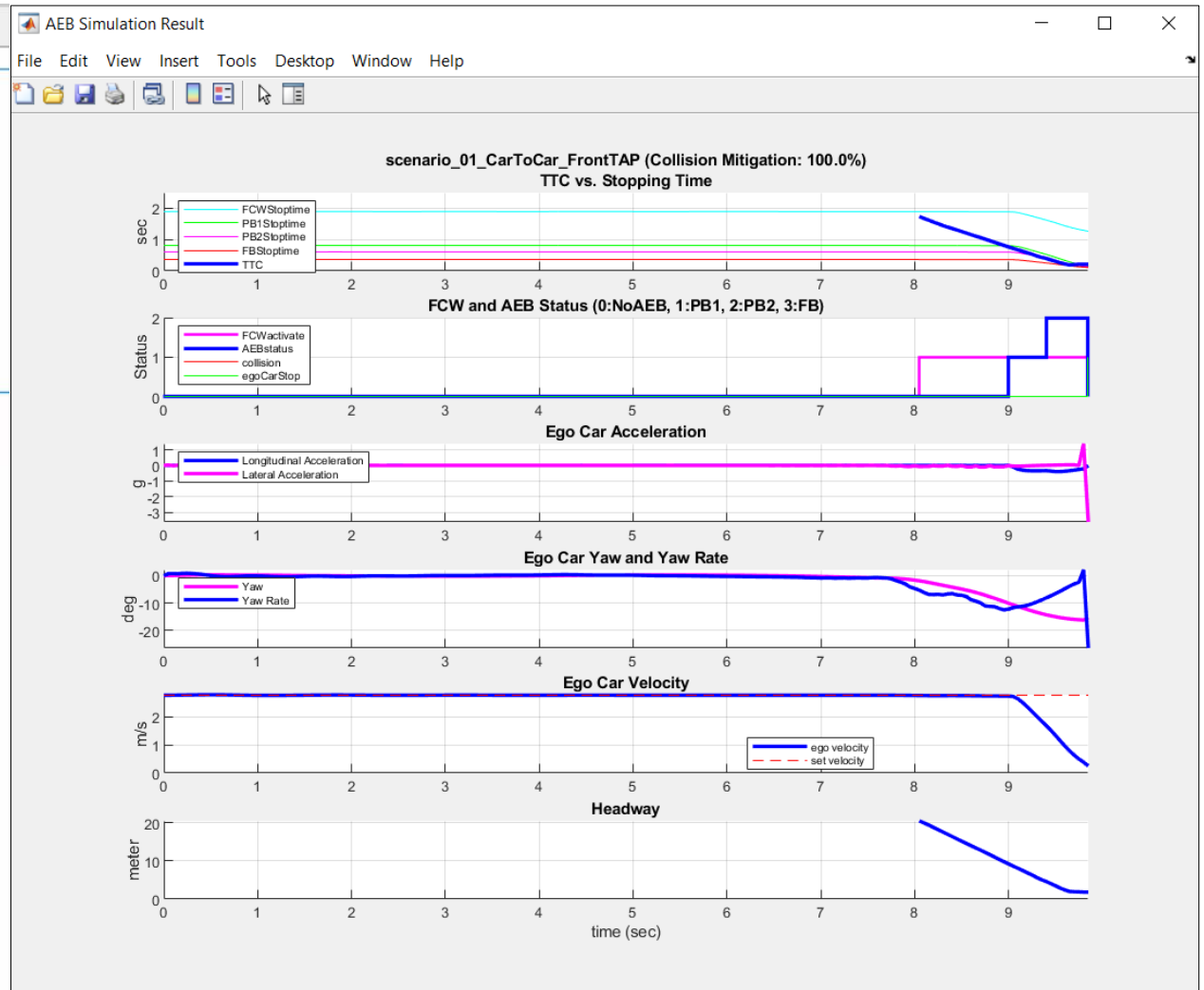
Step 3: Run Simulation

The screenshot displays the MATLAB/Simulink environment during a simulation. The main workspace shows a 2D top-down view of a road intersection with three cars (blue, red, and black) and their respective sensor ranges (radar and vision). The 'AEBWithRRTestBench - Bird's-Eye Scope' window provides a detailed view of the ego vehicle's sensor coverage, including radar and vision cones, and displays various simulation controls like 'Play', 'Step Forward', and 'Stop'. The 'Metrics Assessment' window is overlaid, showing a speedometer for 'Ego Velocity (m/s)' with a needle pointing to approximately 10, a horizontal bar chart for 'Ego Acceleration (m/s^2)' with a value of -0.8, and three status indicators: 'FCW' (grey), 'AEB' (grey), and 'Safety' (green triangle). The 'Controller' window is also visible at the bottom.

Step 3: Run Simulation – Data Postprocessing

```

1 function helperPlotAEBResults(logsout, scenarioFcnName)
2 %helperPlotAEBResults A helper function for plotting the results of the AEB
3 % example
4 % This is a helper function for example purposes and may be removed or
5 % modified in the future.
6 %
7 % The function assumes that the demo outputs the Simulink log, logsout,
8 % containing several signals.
9
10 % Copyright 2018-2022 The MathWorks, Inc.
11
12 if nargin < 2
13     scenarioFcnName = '';
14 end
15
16
17 %% Get the data from simulation
18 ttc = logsout.getElement('TTC'); % TTC : time-to-collision
19 FCWtime = logsout.getElement('FCWstoppingTime'); % stopping times
20 PB1time = logsout.getElement('PB1stoppingTime');
21 PB2time = logsout.getElement('PB2stoppingTime');
22 FBtime = logsout.getElement('FBstoppingTime');
23
24
25 FCWactivate = logsout.getElement('fcw_activate'); % AEB status flags
26 AEBstatus = logsout.getElement('aeb_status');
27 collision = logsout.getElement('collision');
28 egoCarStop = logsout.getElement('ego_car_stop');
29
30 longitudinalAcceleration = logsout.getElement('<ax>'); % longitudinal acceleration of ego car
31 lateralAcceleration = logsout.getElement('<ay>'); % lateral acceleration of ego car
32
33 yaw = logsout.getElement('yaw'); % ego yaw rate
34 yawRate = logsout.getElement('yaw_rate'); % ego yaw rate
35
36 ego_velocity = logsout.getElement('ego_velocity'); % velocity of ego car
37 driver_set_velocity = logsout.getElement('driver_set_velocity'); % driver-set velocity
38
39 headway = logsout.getElement('headway'); % headway between ego car and MIO
40
41 collisionMitigation = logsout.getElement('collisionMitigation'); % collision mitigation
42
43 TTC = ttc.Values.Data;
44 TTC(TTC>1000) = inf;
  
```



Step 4: Scenario Variation – Variables in RoadRunner

The screenshot displays the RoadRunner software interface. The main window shows a 2D editor with a road layout. The road is a T-junction with a vertical road on the left and a horizontal road on the right. A blue car is on the vertical road, and a red car is on the horizontal road. The interface includes a menu bar (File, Edit, View, Tools, Assets, Window, Help) and a toolbar with various icons. The bottom panel is divided into two sections: a logic editor and a variables table.

The logic editor shows a sequence of blocks: a VUT block with a value of 2.8 m/s, followed by a GVT block with a value of 0.0 m/s, and another GVT block with a value of 8.3 m/s. The variables table is empty.

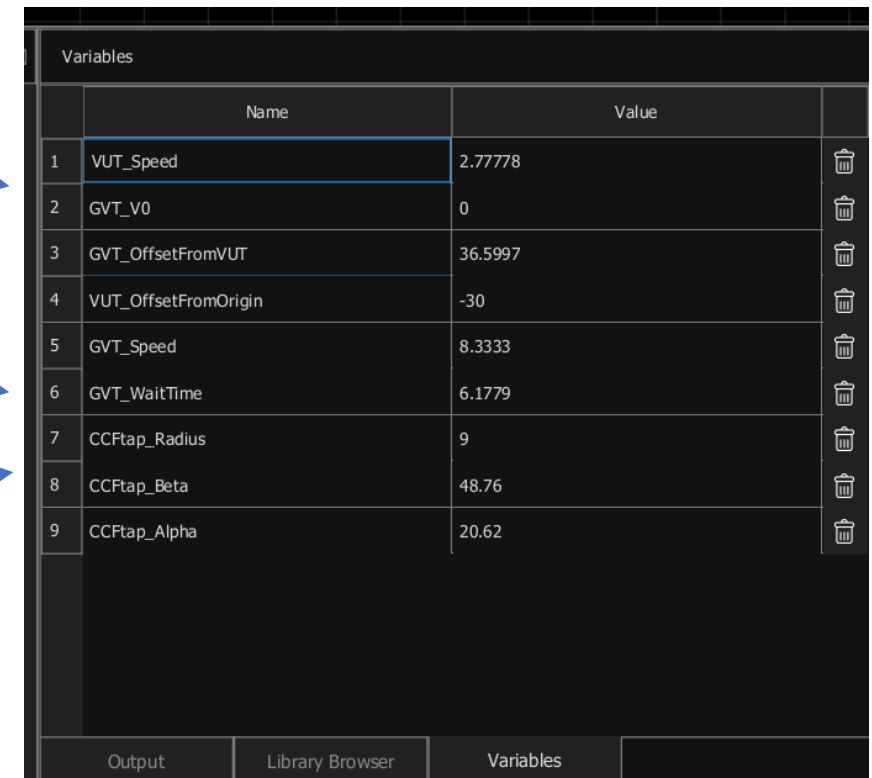
Name	Value
------	-------










Step 4: Scenario Variation – Variables in MATLAB

```

1 %%
2 load("CCFTapVariationData.mat")
3 %%
4 variantID = 9;
5 vutSpeed = table2array(CCFTapVariationData(variantID,1));
6 gvtSpeed = table2array(CCFTapVariationData(variantID,2));
7
8 vutSpeed = vutSpeed * 0.2778; % convert VUT speed from km/hr to m/s
9 gvtSpeed = gvtSpeed * 0.2778; % convert GVT speed from km/hr to m/s
10 rrApp.setScenarioVariable("VUT_Speed",num2str(vutSpeed))
11 rrApp.setScenarioVariable("GVT_Speed",num2str(gvtSpeed))
12 %%
13 % 2. Get the GVT wait time value from |CCFTapVariationData|.
14 %%
15 gvtWaitTime = table2array(CCFTapVariationData(variantID,6));
16 %%
17 % Update the |GVT_WaitTime| variable of RoadRunner Scenario.
18 %%
19 rrApp.setScenarioVariable("GVT_WaitTime",num2str(gvtWaitTime))
20 %%
21 % Based on the VUT speed, set the Euro NCAP path parameters in the
22 % RoadRunner Scenario.
23 %%
24 rrApp.setScenarioVariable("CCFTap_Alpha",num2str(table2array(CCFTapVariationData(variantID,3))))
25 rrApp.setScenarioVariable("CCFTap_Beta",num2str(table2array(CCFTapVariationData(variantID,4))))
26 rrApp.setScenarioVariable("CCFTap_Radius",num2str(table2array(CCFTapVariationData(variantID,5))))
27 %%
28 % 3. Run the setup script to initialize the required variables for the test
29 % bench.
  
```

vutSpeed	gvtSpeed	vutAlpha	vutBeta	vutRadius	gvtWaitTime
10	30	20.62	48.76	9	6.1779
10	45	20.62	48.76	9	7.4019
10	55	20.62	48.76	9	7.847
15	30	20.93	48.14	11.75	2.2
15	45	20.93	48.14	11.75	3.5
15	55	20.93	48.14	11.75	3.9728
20	30	21.79	46.42	14.75	0
20	45	21.79	46.42	14.75	1.3668
20	55	21.79	46.42	14.75	1.8728



Variables			
	Name	Value	
1	VUT_Speed	2.7778	
2	GVT_V0	0	
3	GVT_OffsetFromVUT	36.5997	
4	VUT_OffsetFromOrigin	-30	
5	GVT_Speed	8.3333	
6	GVT_WaitTime	6.1779	
7	CCFTap_Radius	9	
8	CCFTap_Beta	48.76	
9	CCFTap_Alpha	20.62	

Output Library Browser Variables

Step 4: Scenario Variation – Simulink Test Manager

Test Manager

TESTS

New Open Save Cut Copy Delete Test Spec Report Run

Test File
Create a blank test file

Test Suite
Create a container for test cases

MATLAB-based Simulink Test
Create a blank MATLAB-based Simulink Test file

TEST CASE TEMPLATES

Baseline Test
Compare simulation output to a fixed baseline signal

Equivalence Test
Compare output of two simulations

Simulation Test
Perform a simulation with no criteria

Real-Time Test
Perform a simulation on real-time target

RoadRunner Test
Perform driving scenario simulation using RoadRunner

Test Manager

TESTS

New Open Save Cut Copy Delete Test Spec Report Run Run with Stepper Stop Parallel Report Visualize Highlight in Model Import Export Model Testing Dashboard Preferences Help

FILE EDIT RUN RESULTS ENVIRONMENT RESOURCES

Test Browser Results and Artifacts

CarToCarFrontTapTestIterations Start Page

Filter tests by name or tags, e.g. tags: test

AutomateCCFTapScenarioVariationTests

CCFTap

CarToCarFrontTapTestIterations

CarToCarFrontTapTestIterations Enabled

AutomateCCFTapScenarioVariationTests > CCFTap > CarToCarFrontTapTestIterations

RoadRunner Test (Simulation Test)

TAGS

DESCRIPTION

REQUIREMENTS

No requirements available. Click on Add button to open the link editor or the drop down to explore other options.

ROADRUNNER*

SYSTEM UNDER TEST*

PARAMETER OVERRIDES

CALLBACKS*

INPUTS

SIMULATION OUTPUTS

CONFIGURATION SETTINGS OVERRIDES

ITERATIONS*

TABLE ITERATIONS

SCRIPTED ITERATIONS*

Help on creating test iterations:

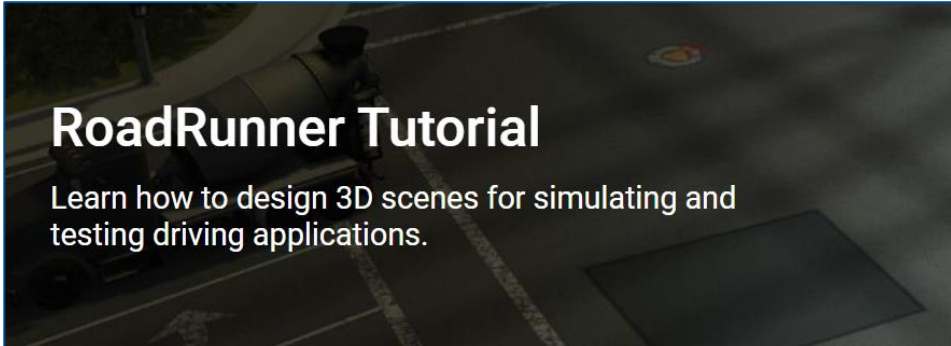
```

1 % Load predefined CCFTap variation data from mat file
2 load("CCFTapVariationData", "CCFTapVariationData");
3
4 % Create iterations
5 for i = 1:size(CCFTapVariationData,1)
6
7     % Create the test iteration object
8     testIter = sltestiteration;
9
10    % Set VUT and GVT Speed
11    vutSpeed = table2array(CCFTapVariationData(i,1));
12    gvtSpeed = table2array(CCFTapVariationData(i,2));
13
14    vutSpeed = vutSpeed * 0.2778; % convert VUT speed from km/hr to m/s
15    gvtSpeed = gvtSpeed * 0.2778; % convert GVT speed from km/hr to m/s
16
17    % Overwrite the VUT and GVT speed variables available in the RoadRunner scenario
  
```

PROPERTY VALUE

PROPERTY	VALUE
Name	CarToCarFrontTapTestIter
Type	RoadRunner Test (Simulati...
Model	AEBWithIRRTestBench
Simulation Mode	[Model Settings]
Location	C:\Users\Mateusz\Docume...
Enabled	<input checked="" type="checkbox"/>
Hierarchy	AutomateCCFTapScenario...
Tags	Type comma or space separa...

RoadRunner Online Training



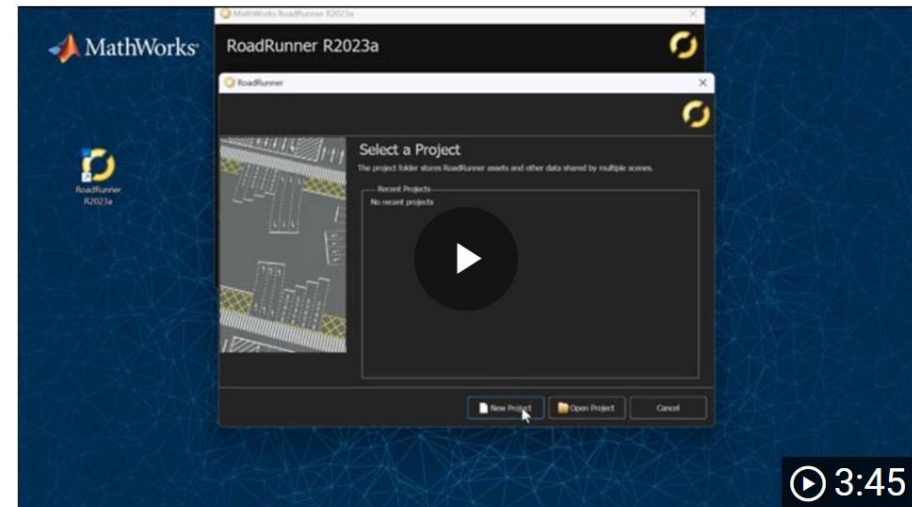
RoadRunner Tutorial
Learn how to design 3D scenes for simulating and testing driving applications.

1. Launching RoadRunner	5. Customizing Lanes
2. Using Camera Controls	6. Creating Junctions
3. Exporting Scenes to Simulators	7. Adding Terrains and Props
4. Creating Roads	8. Conclusion



1. Launching RoadRunner

RoadRunner is an interactive editor that lets you design 3D scenes for simulating and testing automated driving systems. This module describes how to launch RoadRunner, create a new project and scene, and identify the different panes in the RoadRunner interface.



Summary

- Launch RoadRunner and create new projects and scenes.
- Load prebuilt scenes.
- Identify the toolstrip and other panes in the RoadRunner interface.



Oprogramowanie
Naukowo-Techniczne
sp. z o.o.

www.ont.com.pl



matlab.pl



oprogramowanie-
naukowo-techniczne



ONT MATLAB

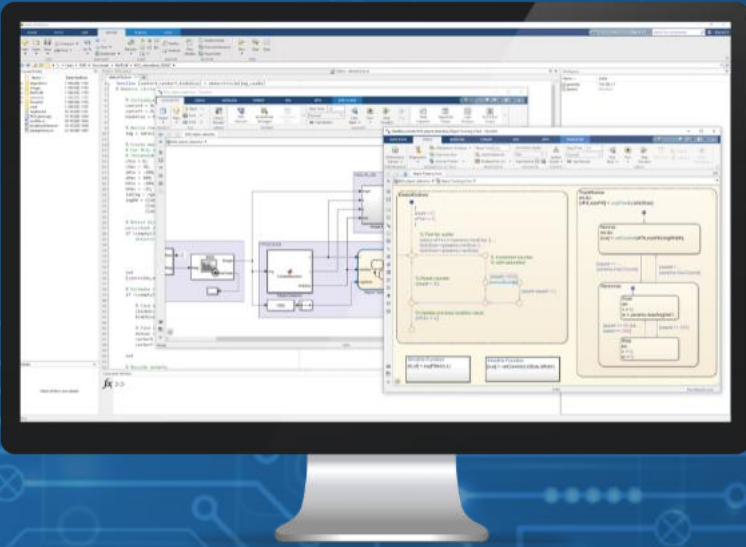


Mateusz Łabęcki

Application Engineer, ONT

mateusz.labecki@ont.com.pl

tel. 534 884 988



APPLICATIONS

- ▶ Robotics and Automation
- ▶ Computational Finance
- ▶ Autonomous Vehicles
- ▶ Electronics
- ▶ Artificial Intelligence
- ▶ Biomedical Engineering
- ▶ Systems Engineering and certification
- ▶ Power Electronics and Systems
- ▶ Communications and Radar Systems

Let's stay in touch

Oprogramowanie Naukowo-Techniczne sp. z o.o.
MATLAB and Simulink authorised reseller for Poland
ul. Pod Fortem 19, 31-302 Kraków, Poland | www.ont.com.pl