
# MathWorks Toolchain for Low-Velocity Maneuvering Development at General Motors

Alon Davidi

Jonathan Naor

The last hundred meters of an autonomous drive entail unique challenges: there is variability in direction (reverse, three-point turns, etc.) there are often no road markings, no GPS signal, and no map.

This is where a Self-Driving Car is at its most autonomous.


The goal of the Low Velocity Maneuvering (LVM) team at General Motors is to drive the vehicle in GPS-denied environments with high accuracy, to enable a variety of autonomous features.

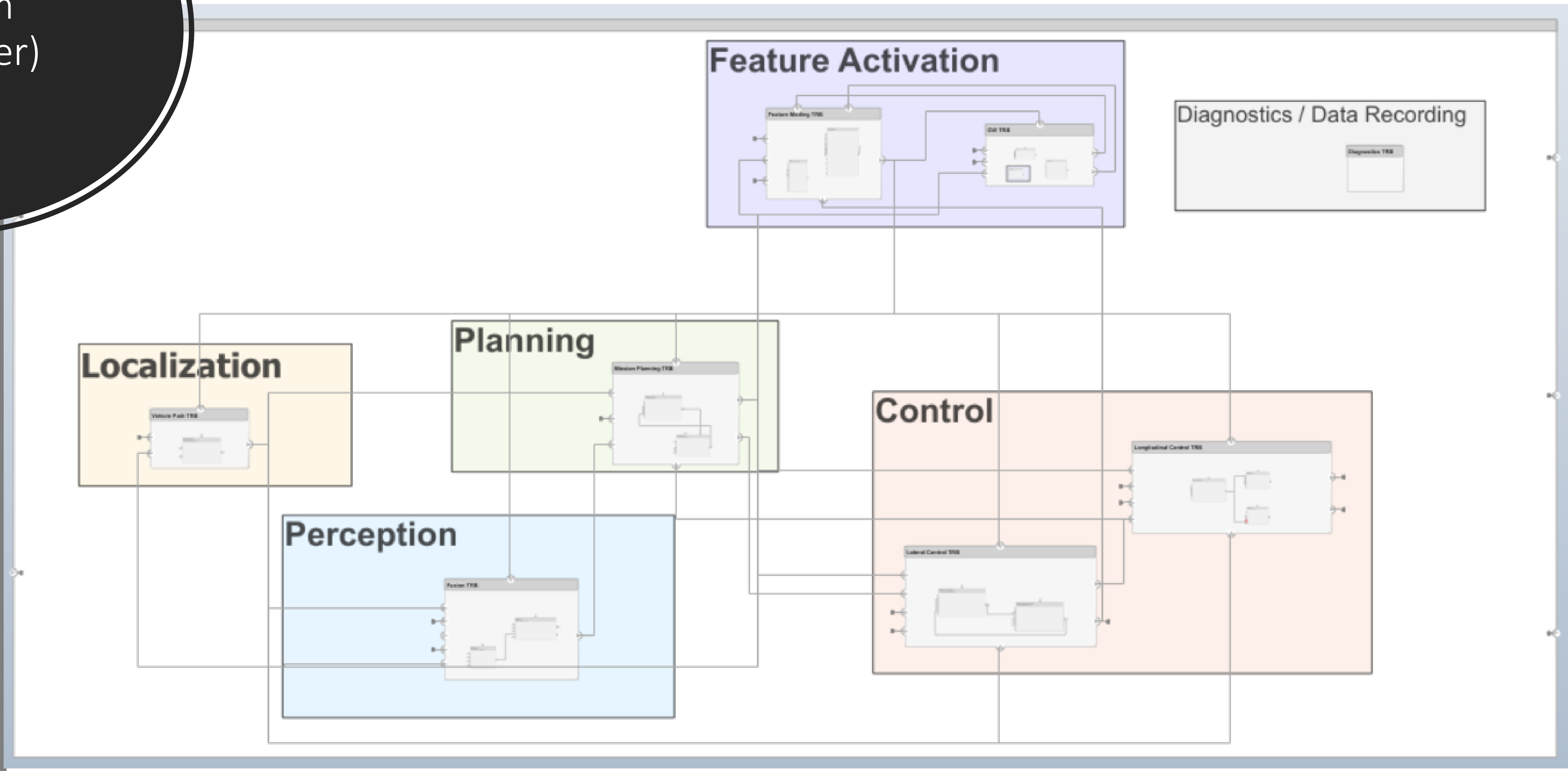
In this talk we review our LVM development cycle, emphasizing MathWorks tools utilization, starting from the architecture management, models base design development, requirements & testing coverage, MIL, SIL, HIL simulations and code generation for multiple platforms.


# Development Process Motivation

- Fast development cycle: fast transition from proof of concept to production-mature software
- Utilize the best of all worlds for simulation, development, validation and deployment:
  - **CarSim** for precise vehicle dynamics
  - **Unreal**-based fisheye photo-realistic image rendering
  - **ROS2** for visualization and debugging
  - **dSpace** for real-time validation
  - **Simulink** for Model-based design

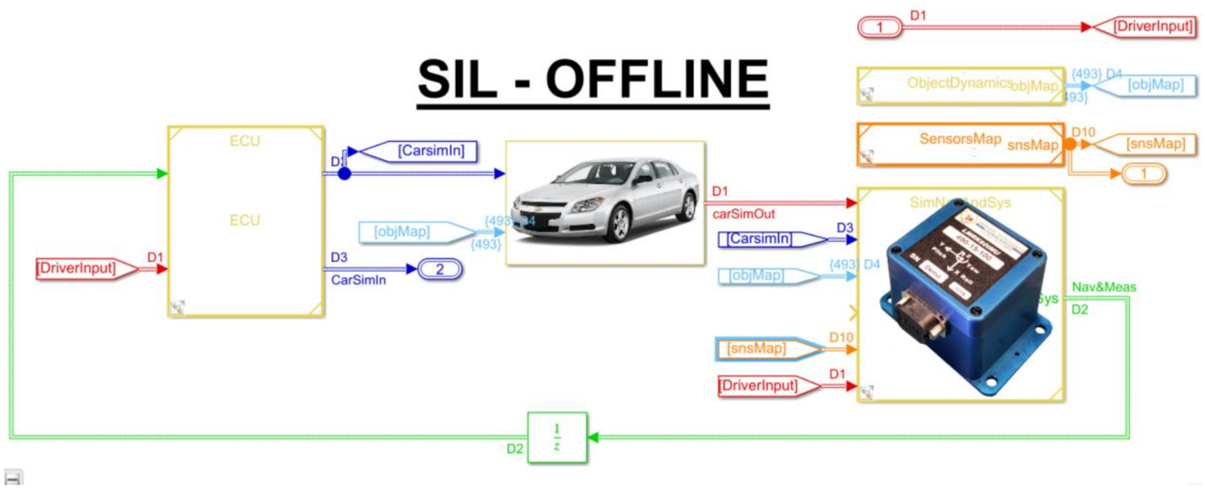
# Architecture management (System Composer)

Easy System and Architecture management – Model-based design approach

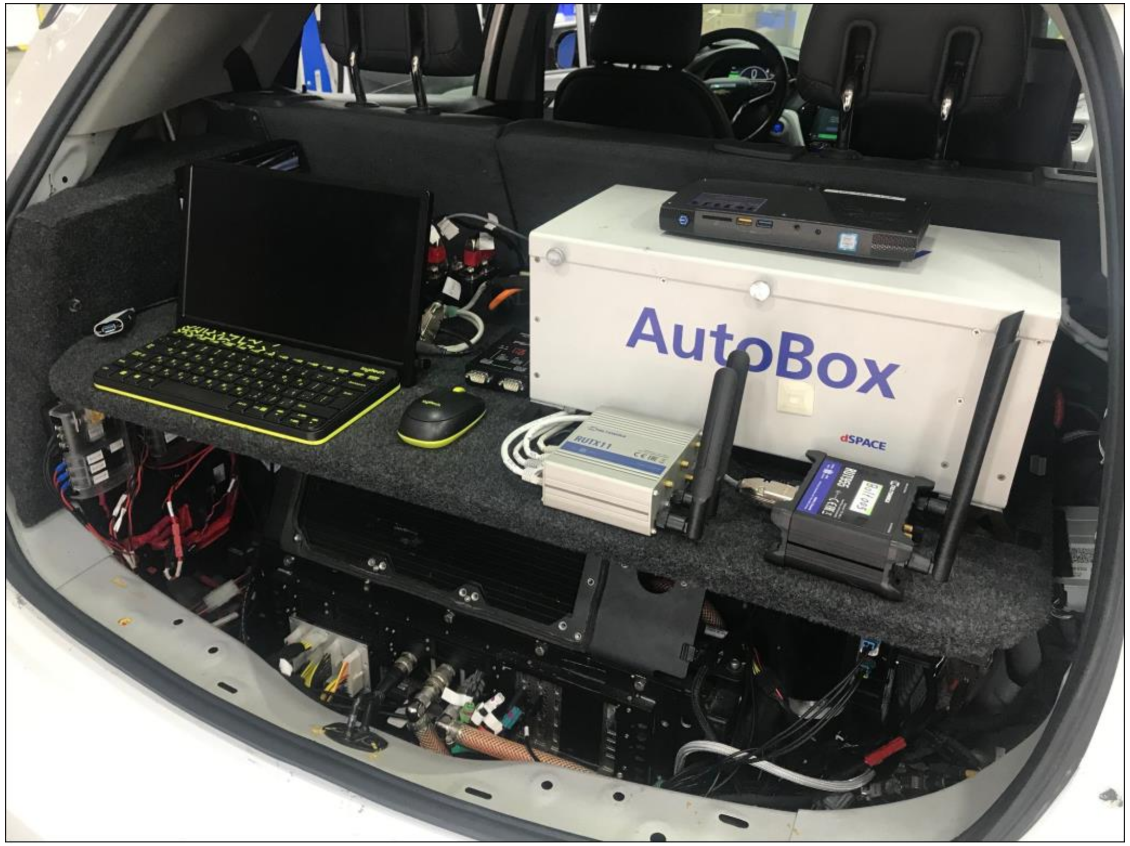


# Simulink/CarSim co-simulation (SIL)

In-house full vehicle system simulation

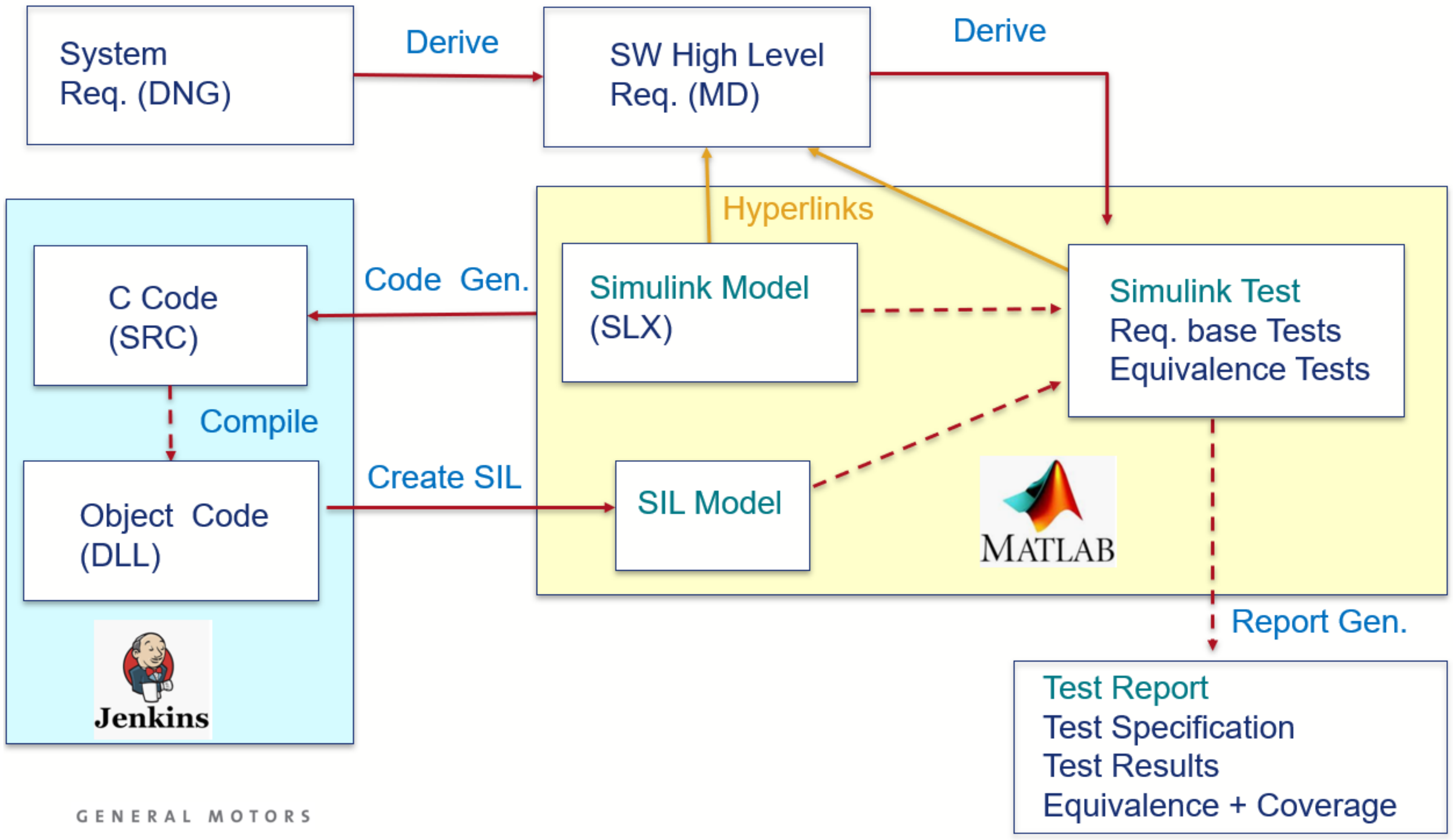


# RT HIL/Vehicle Validation –
# co-simulation (Simulink, Dspace & CarSim / Vehicle)

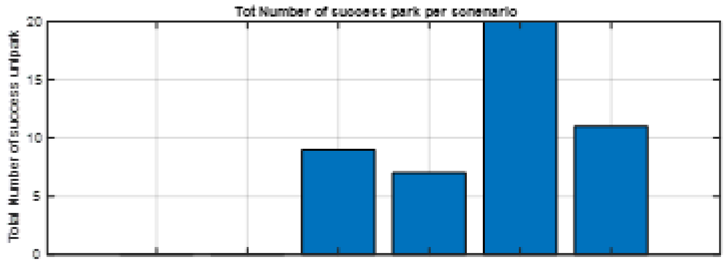
Model and Code validation – “white box”

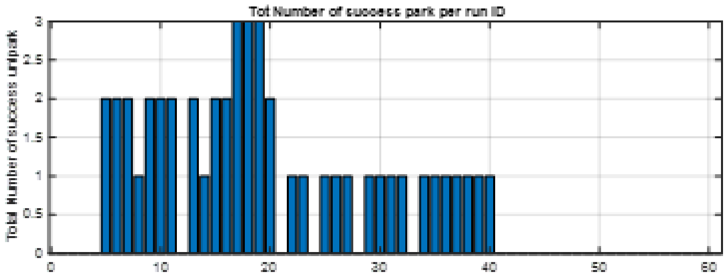
Unit-test process flowchart using Test Manager for requirements coverage analysis

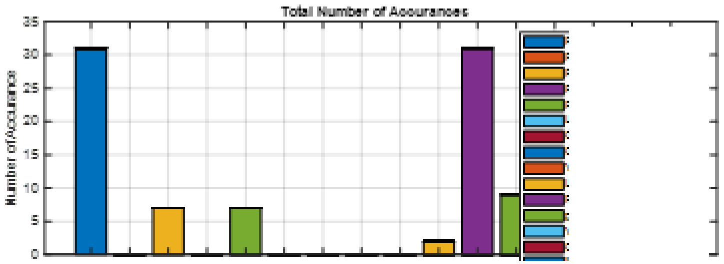


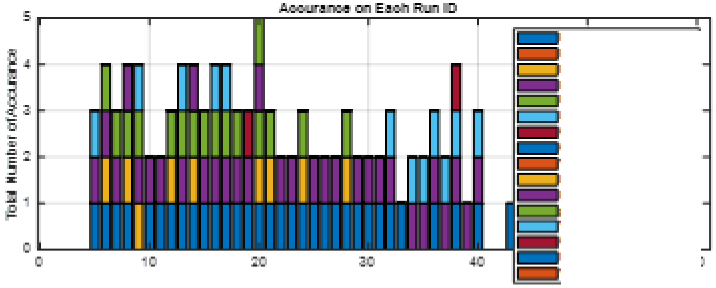
# Model Validation “Black box” (Monte-Carlo)

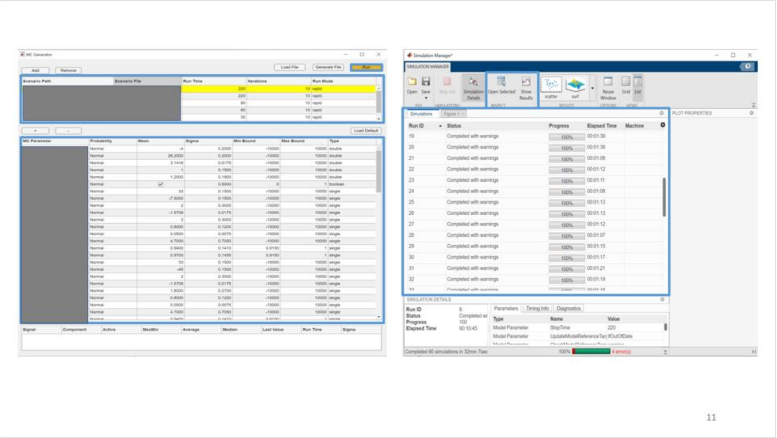
System-level module in the loop verification using Simulation Manager

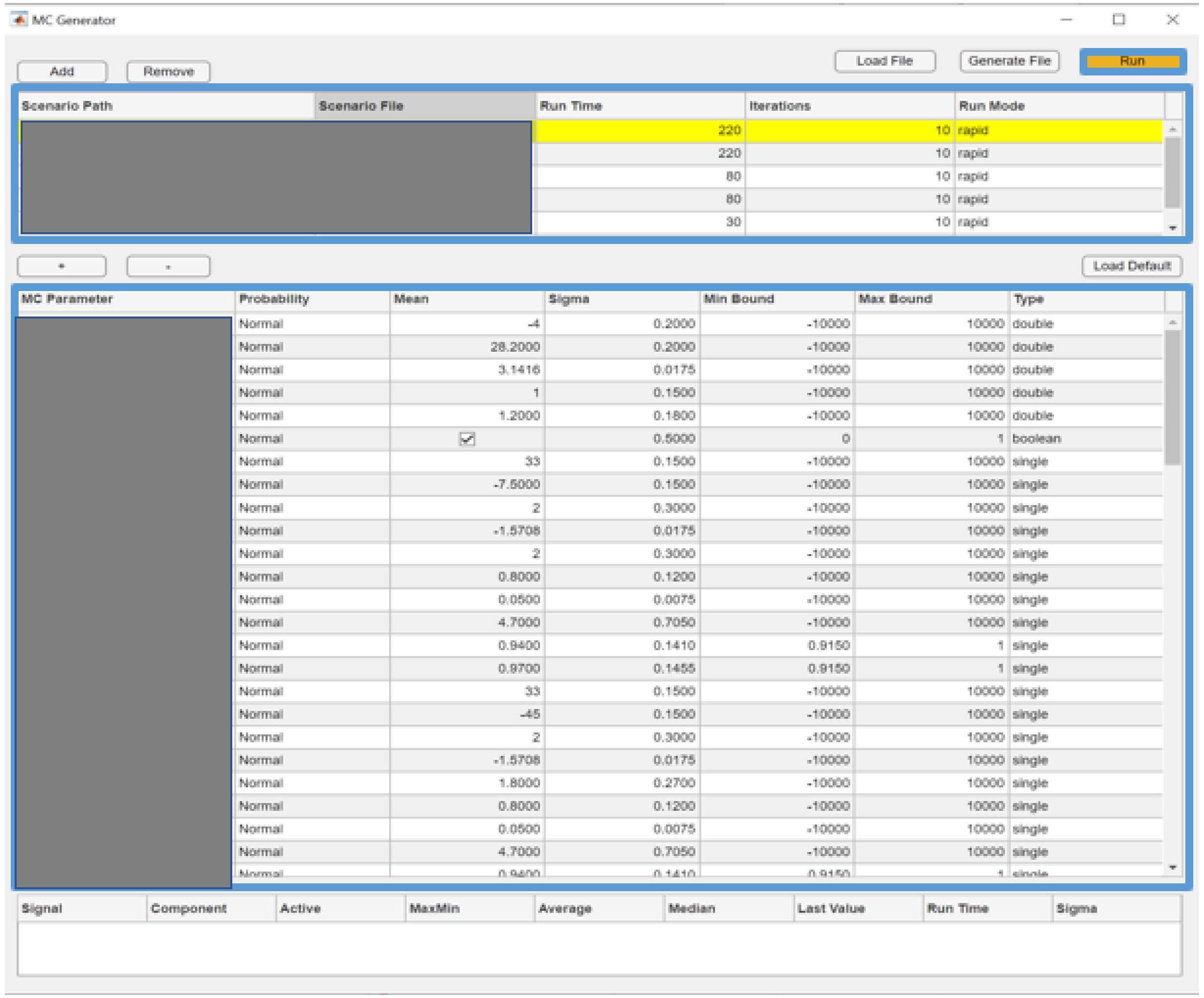
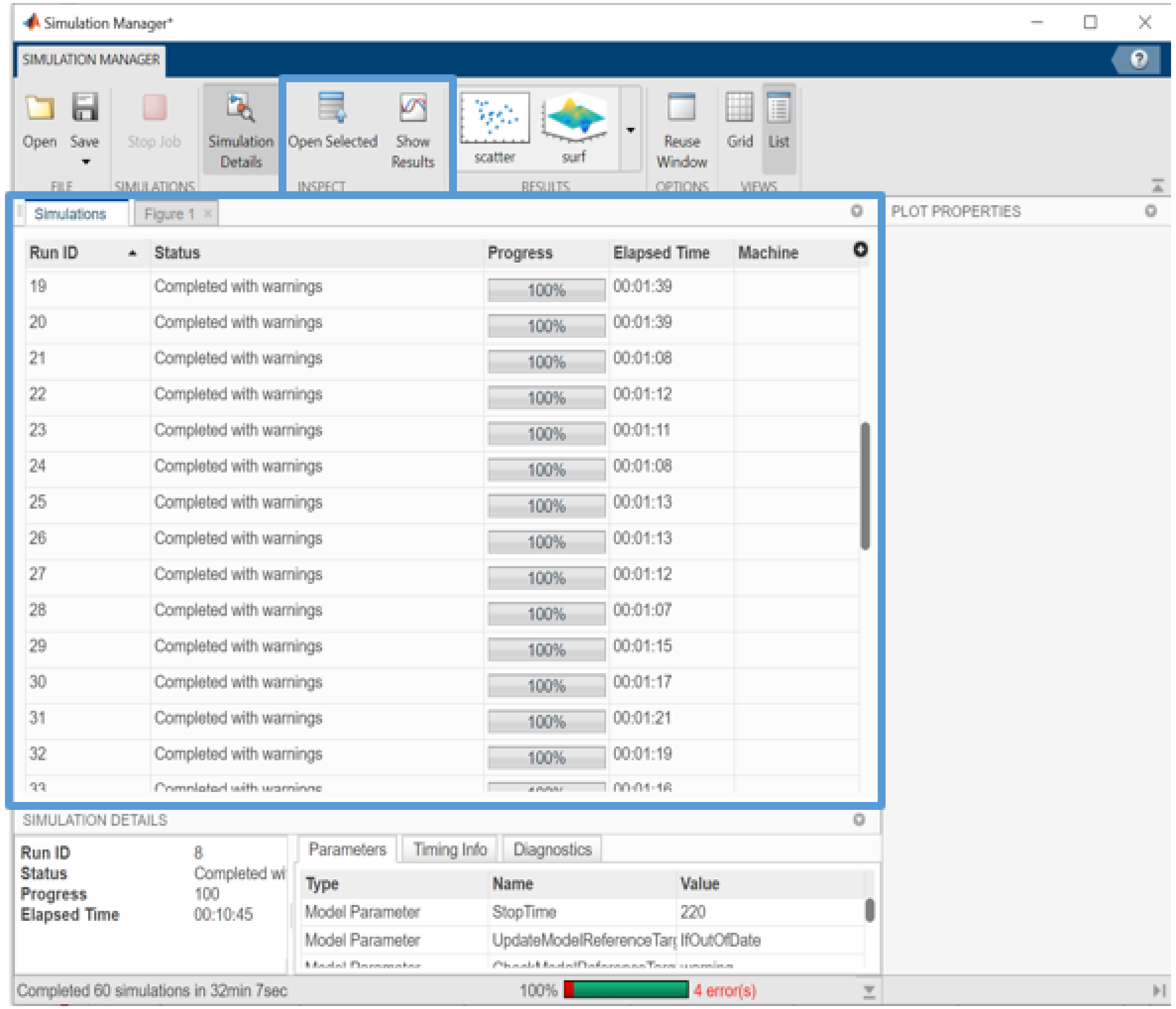


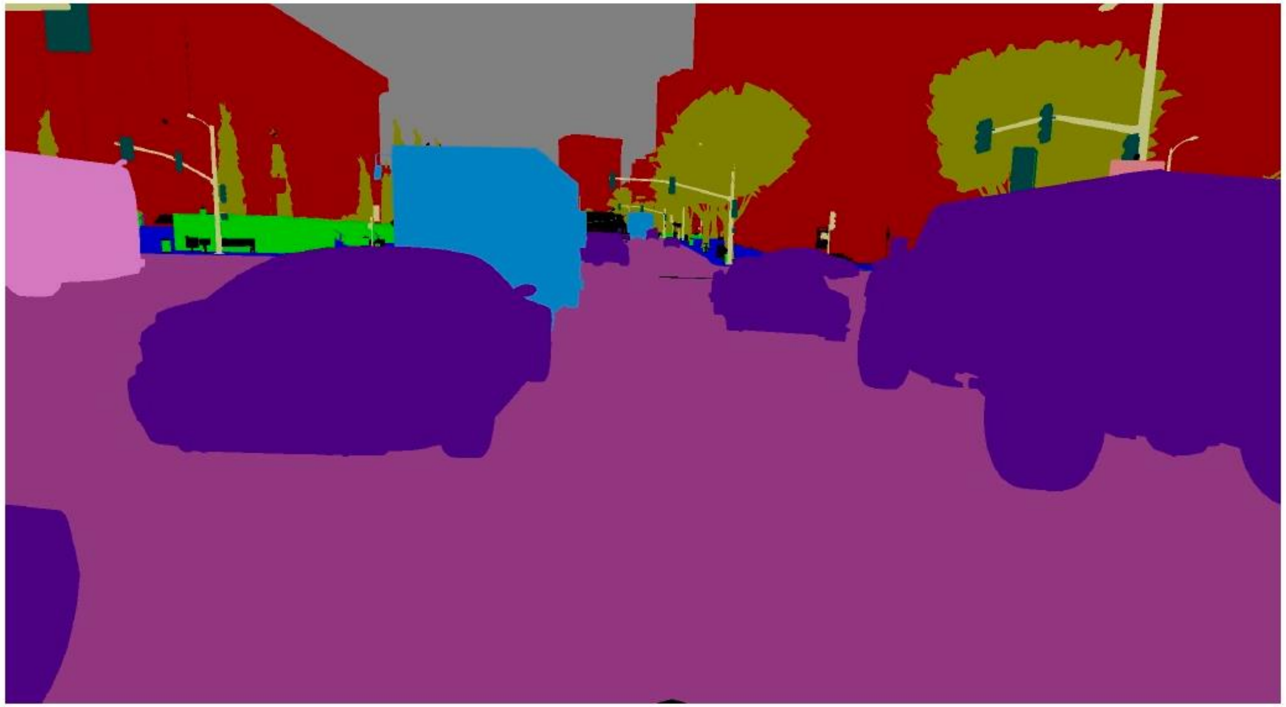

MC Generator
Add
Remove
Load File
Generate File
Run
Scenario Path
Scenario File
Run Time
Iterations
Run Mode
220 10 rapid
220 10 rapid
80 10 rapid
80 10 rapid
30 10 rapid
+
-
Load Default
MC Parameter
Probability
Mean
Sigma
Min Bound
Max Bound
Type
Normal -4 0.2000 -10000 10000 double
Normal 28.2000 0.2000 -10000 10000 double
Normal 3.1416 0.0175 -10000 10000 double
Normal 1 0.1500 -10000 10000 double
Normal 1.2000 0.1800 -10000 10000 double
Normal 0.5000 0 1 boolean
Normal 33 0.1500 -10000 10000 single
Normal -7.5000 0.1500 -10000 10000 single
Normal 2 0.3000 -10000 10000 single
Normal -1.5708 0.0175 -10000 10000 single
Normal 2 0.3000 -10000 10000 single
Normal 0.8000 0.1200 -10000 10000 single
Normal 0.0500 0.0075 -10000 10000 single
Normal 4.7000 0.7050 -10000 10000 single
Normal 0.9400 0.1410 0.9150 1 single
Normal 0.9700 0.1455 0.9150 1 single
Normal 33 0.1500 -10000 10000 single
Normal -45 0.1500 -10000 10000 single
Normal 2 0.3000 -10000 10000 single
Normal -1.5708 0.0175 -10000 10000 single
Normal 1.8000 0.2700 -10000 10000 single
Normal 0.8000 0.1200 -10000 10000 single
Normal 0.0500 0.0075 -10000 10000 single
Normal 4.7000 0.7050 -10000 10000 single
Signal
Component
Active
MaxMin
Average
Median
Last Value
Run Time
Sigma

Simulation Manager*
SIMULATION MANAGER
Open
Save
Stop Job
Simulation Details
Open Selected
Show Results
scatter
surf
Reuse Window
Grid
List
FILE
SIMULATIONS
INSPECT
RESULTS
OPTIONS
VIEWS
Simulations
Figure 1
PLOT PROPERTIES
Run ID
Status
Progress
Elapsed Time
Machine
19 Completed with warnings 100% 00:01:39
20 Completed with warnings 100% 00:01:39
21 Completed with warnings 100% 00:01:08
22 Completed with warnings 100% 00:01:12
23 Completed with warnings 100% 00:01:11
24 Completed with warnings 100% 00:01:08
25 Completed with warnings 100% 00:01:13
26 Completed with warnings 100% 00:01:13
27 Completed with warnings 100% 00:01:12
28 Completed with warnings 100% 00:01:07
29 Completed with warnings 100% 00:01:15
30 Completed with warnings 100% 00:01:17
31 Completed with warnings 100% 00:01:21
32 Completed with warnings 100% 00:01:19
SIMULATION DETAILS
Run ID 8
Status Completed wi
Progress 100
Elapsed Time 00:10:45
Parameters
Timing Info
Diagnostics
Type
Name
Value
Model Parameter StopTime 220
Model Parameter UpdateModelReferenceTarg IfOutOfDate
Completed 60 simulations in 32min 7sec
100%
4 error(s)

# Adding vision perception to system simulation

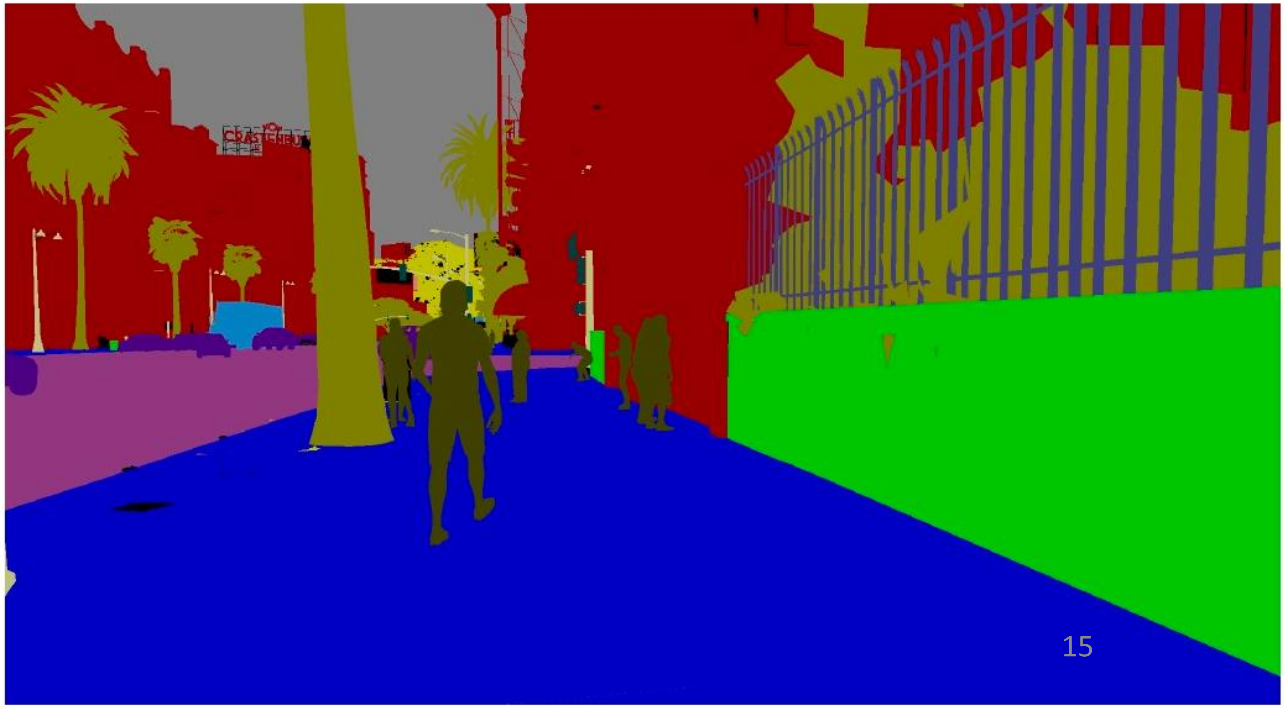
Photorealism

# Requirements: Fisheye Image Generation


# Requirements: Simulated Weather Effects

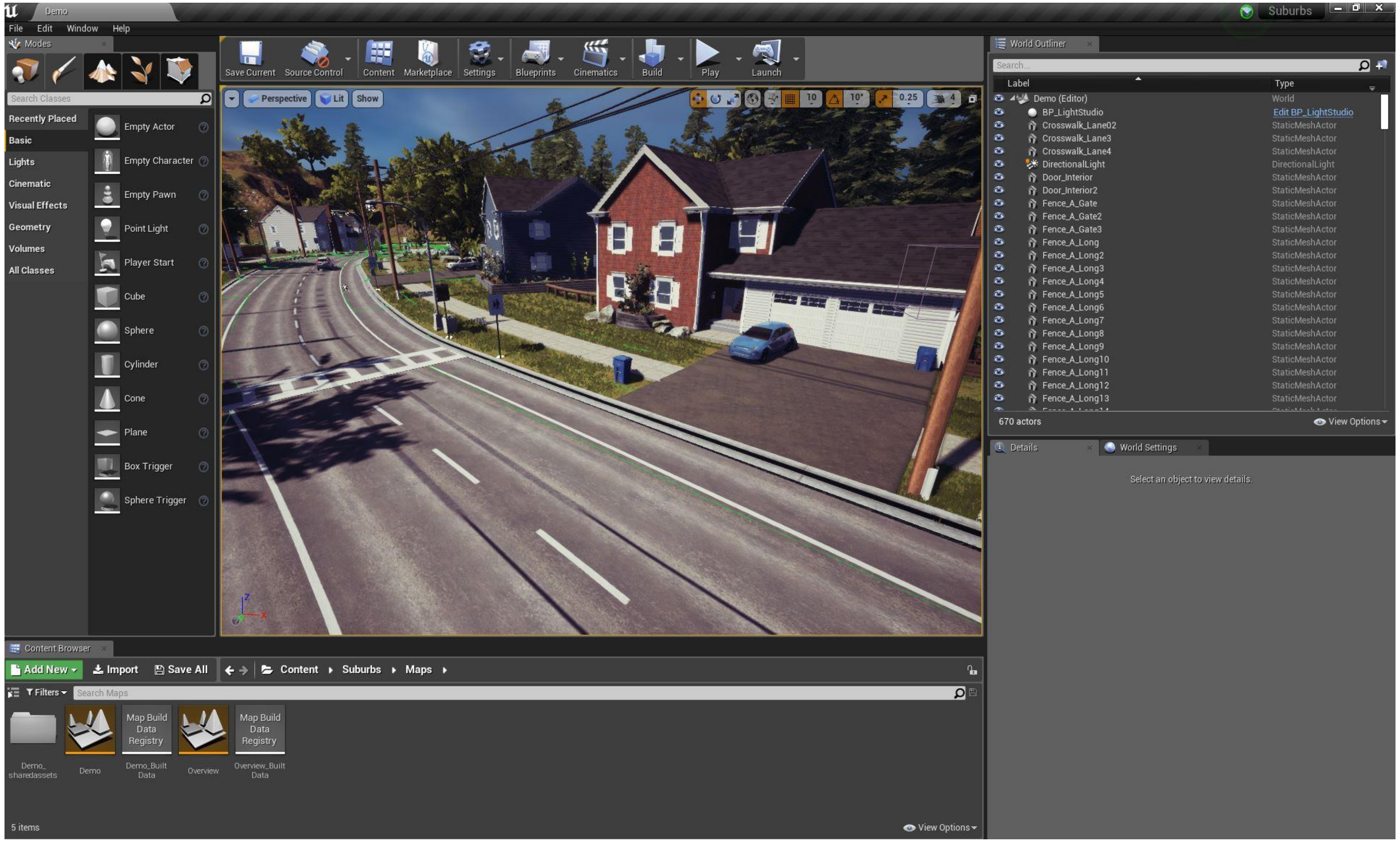


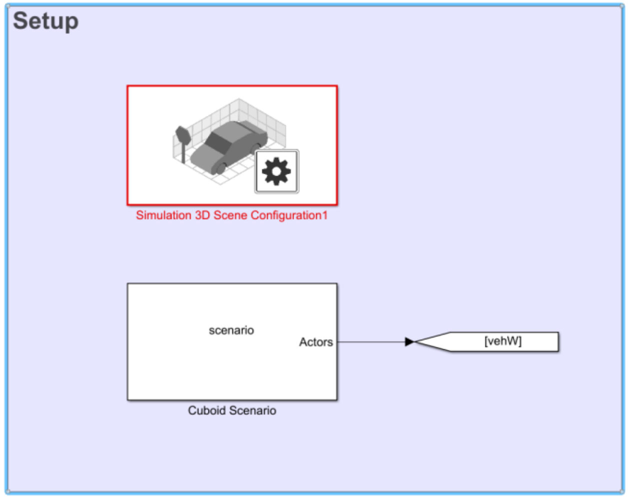
# Requirements: Ground Truth

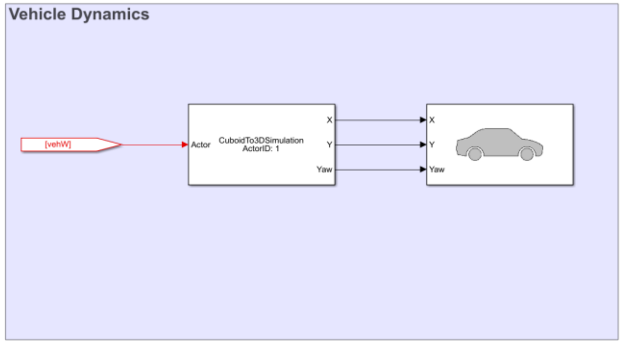


# Requirements: Recording and Visualization

# Photorealism (Unreal Engine Interface)

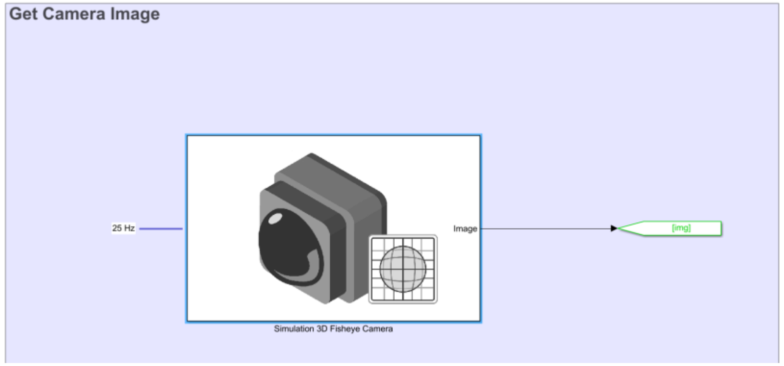


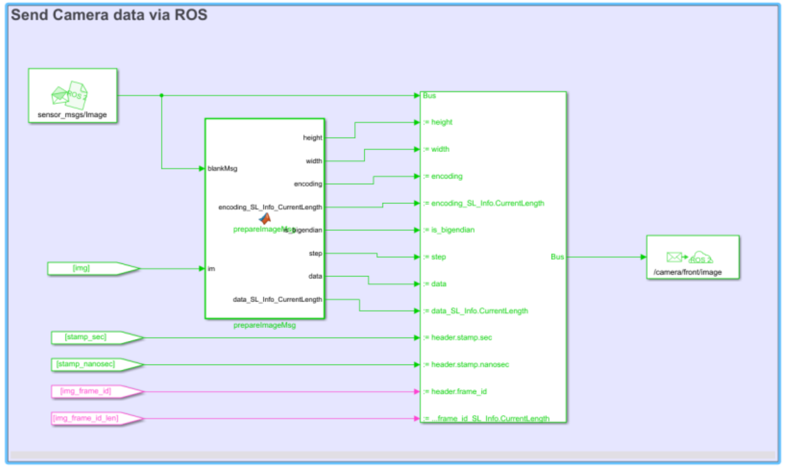




Interface with Unreal Engine enables state of the art photorealistic images in a variety of scenarios

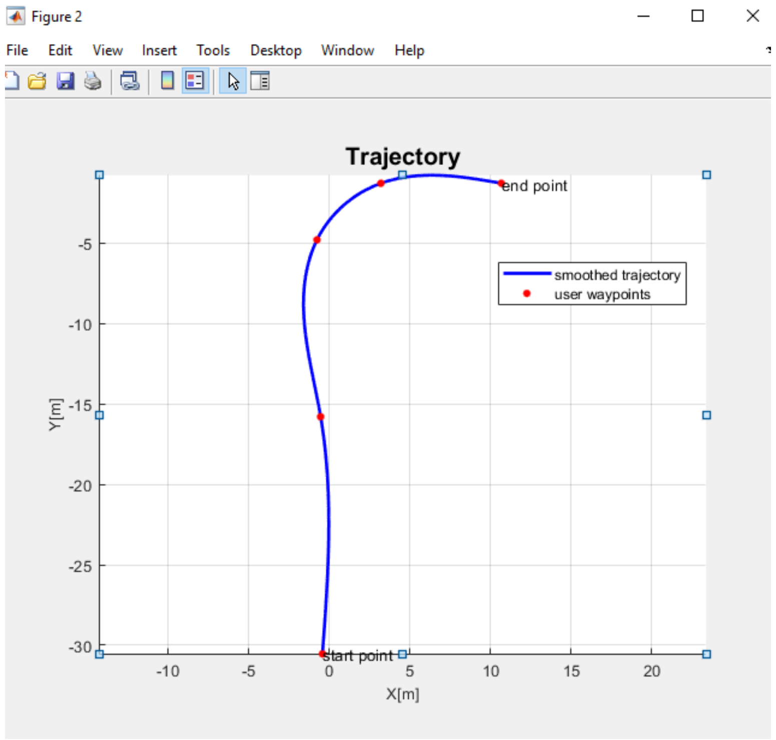
# Fisheye Image Generation



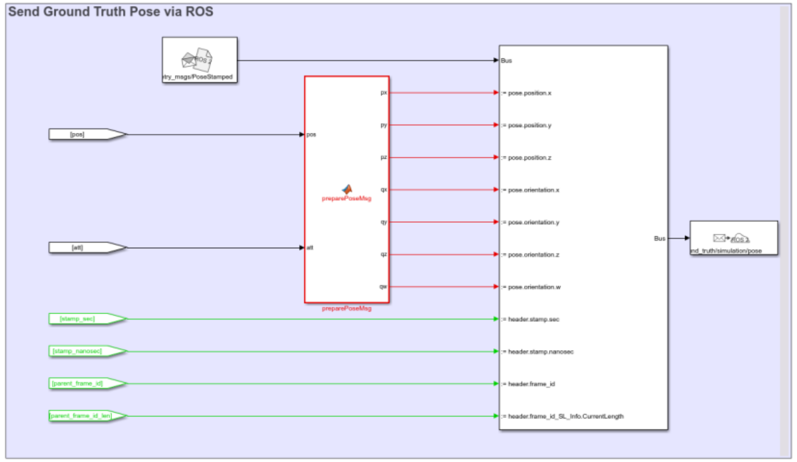




Unique ability to generate fisheye images from Unreal
(configurable intrinsics and extrinsics)

# Ground Truth

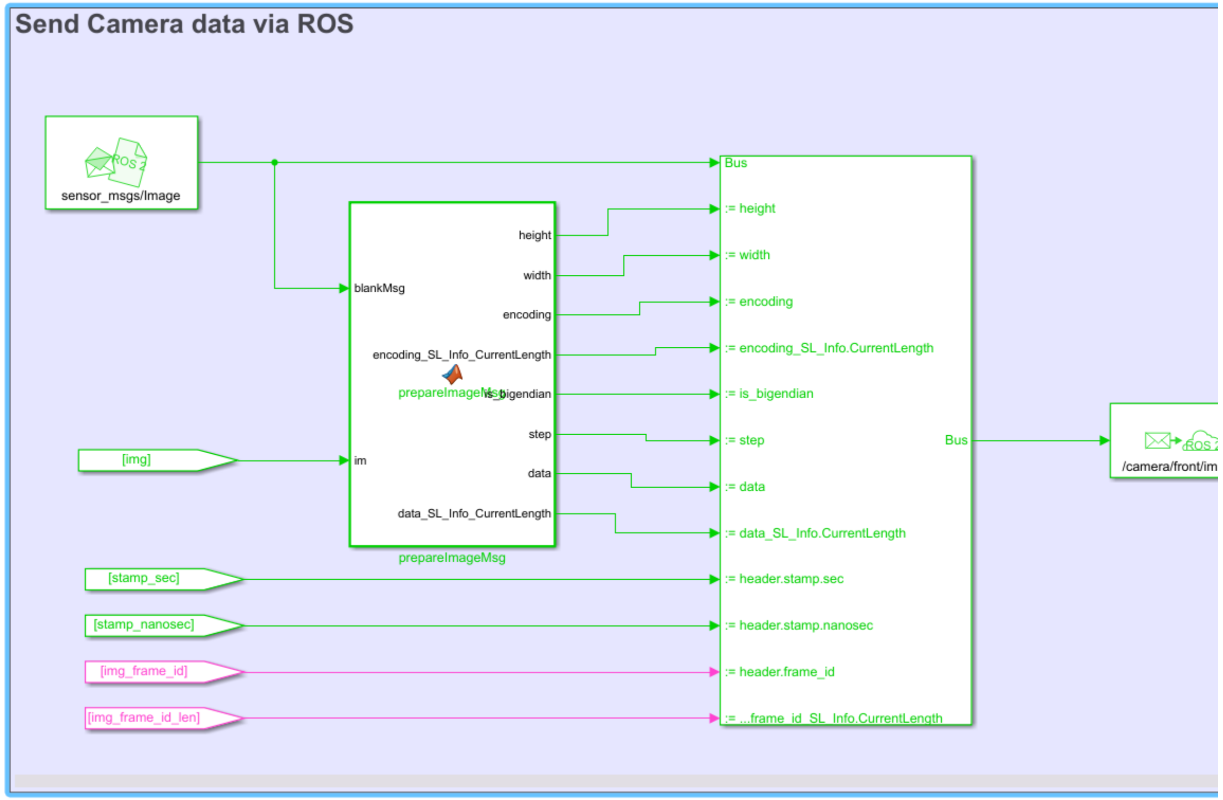


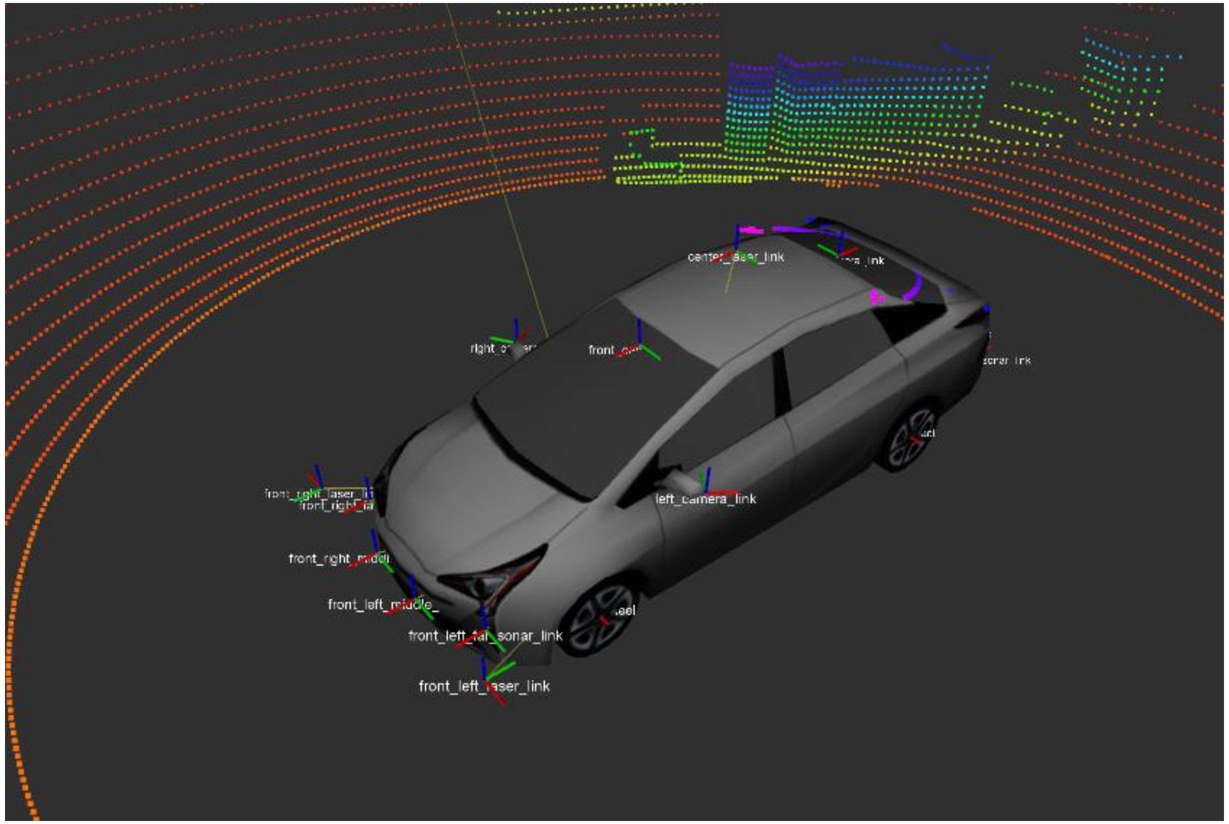




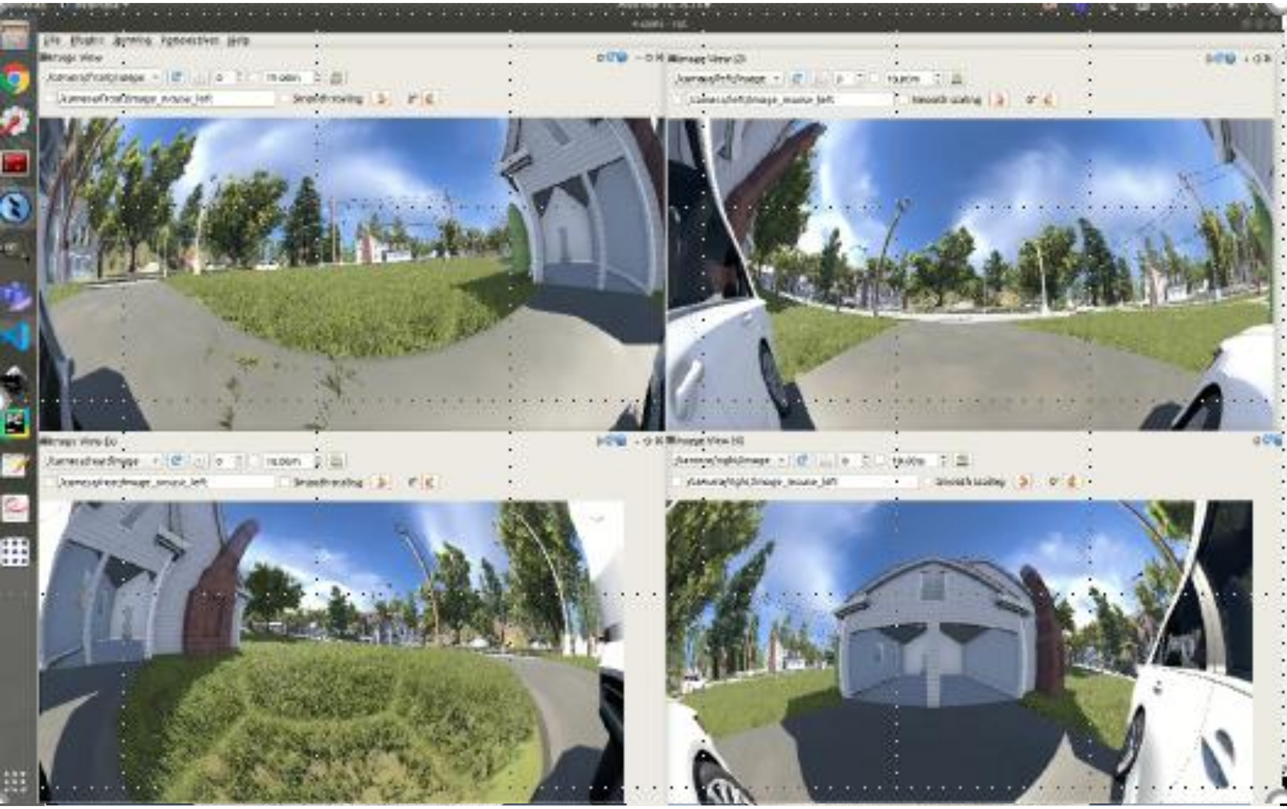

Generate route in MATLAB / Simulink and execute in Unreal

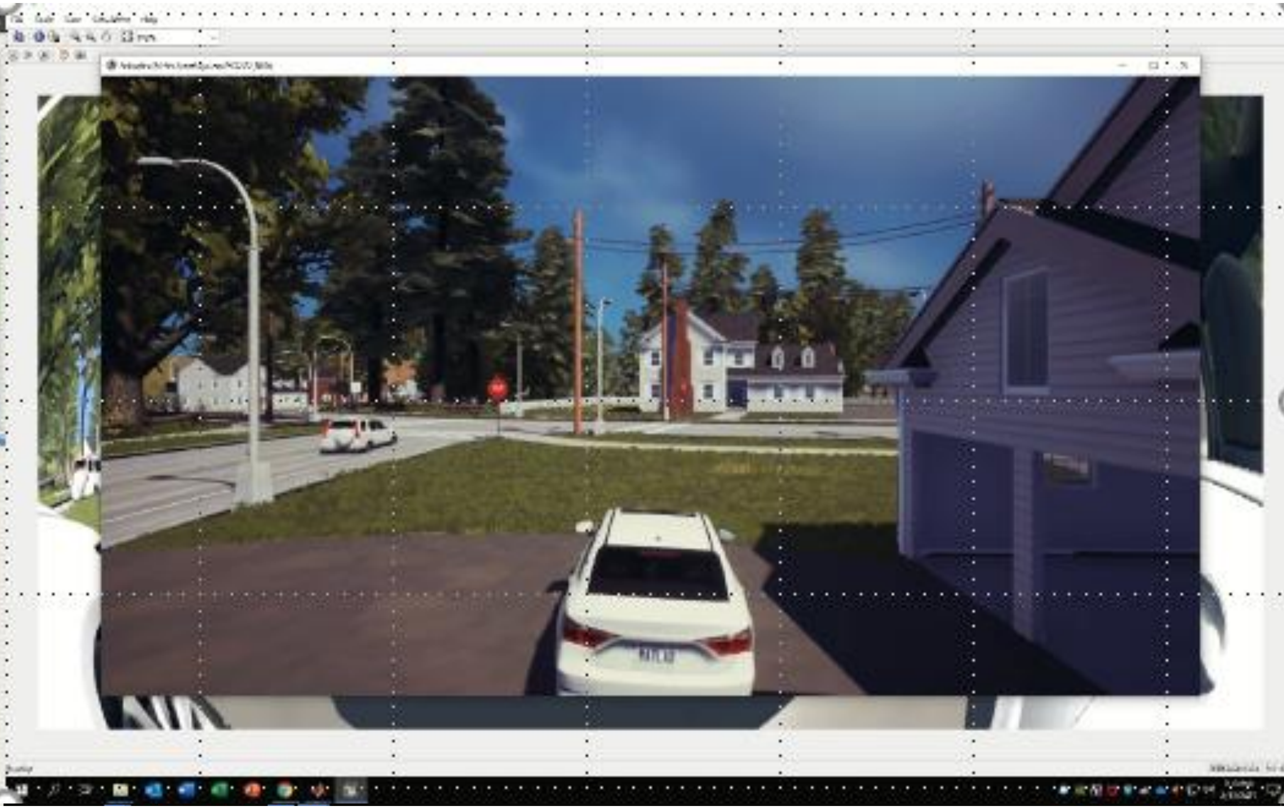
# Recording & Visualization

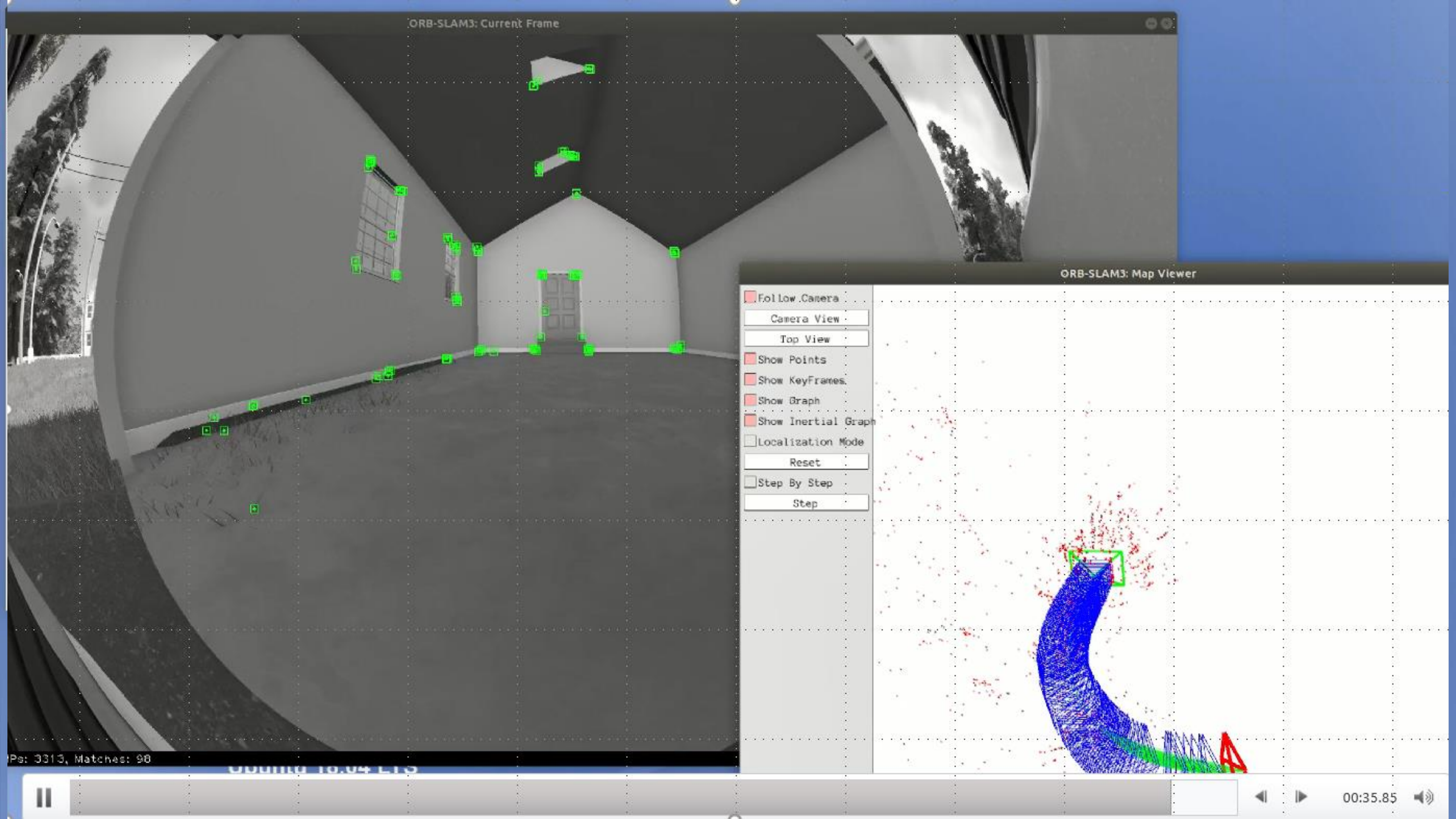



Broadcast sensor data and ground truth in standard ROS2 messages; utilize debugging and visualization tools






ORB-SLAM3: Current Frame
Ps: 3313, Matches: 98
ORB-SLAM3: Map Viewer
Follow Camera
Camera View
Top View
Show Points
Show KeyFrames
Show Graph
Show Inertial Graph
Localization Mode
Reset
Step By Step
Step
00:35.85

# Wrap up

Using MATLAB/Simulink we were able to quickly:

- Manage
    - Architecture management using System Composer
- Develop
    - Control System
    - Vision Perception System
- Validate
    - Requirement-based and back-to-back testing using Simulink Test
- Deploy
    - Code generation

**Next Steps:**

- Unify control and vision development platforms
- Move to Adaptive AutoSAR architecture

# MathWorks and GM Collaborative Effort

Thank you!