



MATLAB®  
& SIMULINK®

# MATLAB/Simulinkによる自動運転アルゴリズム開発とシミュレーション ～センサーフュージョンを題材として～

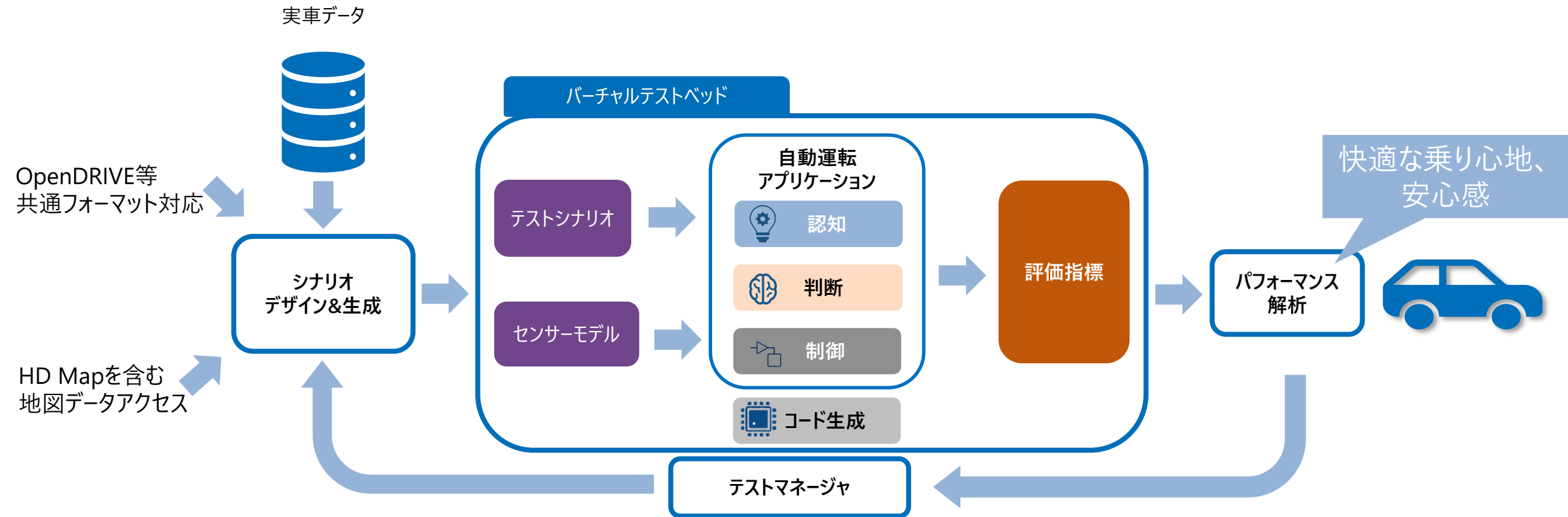
MathWorks Japan

大塚 慶太郎

kotsuka@mathworks.com



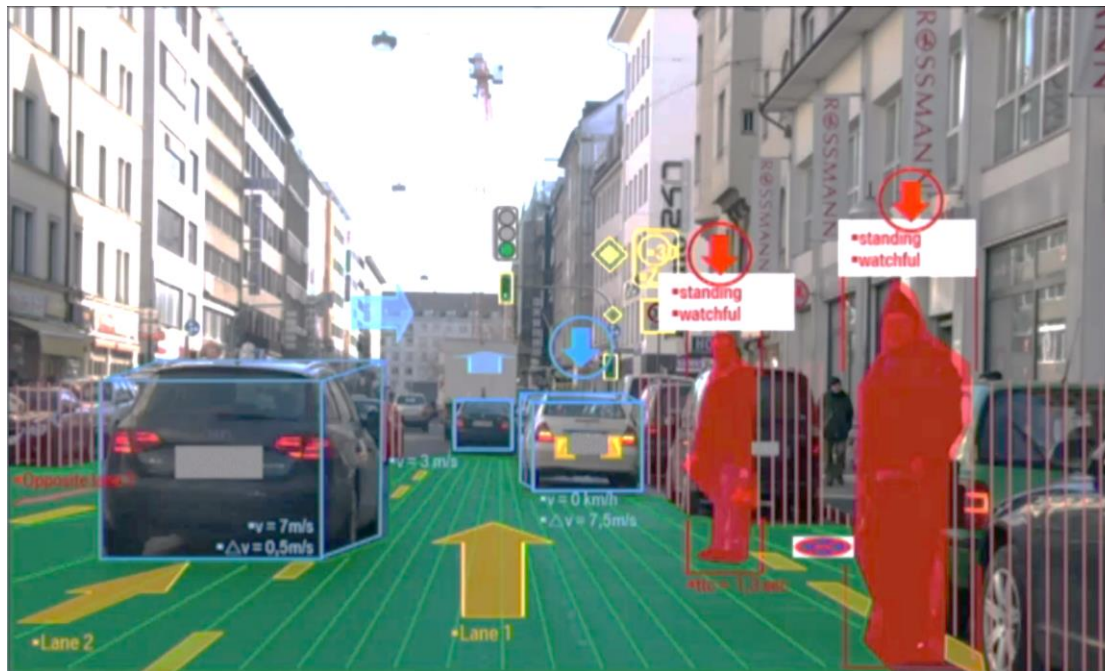
# ADAS/自動運転システム 統合開発環境



# ADAS・自動運転システムの開発 – 認知 with MATLAB and Simulink



認知

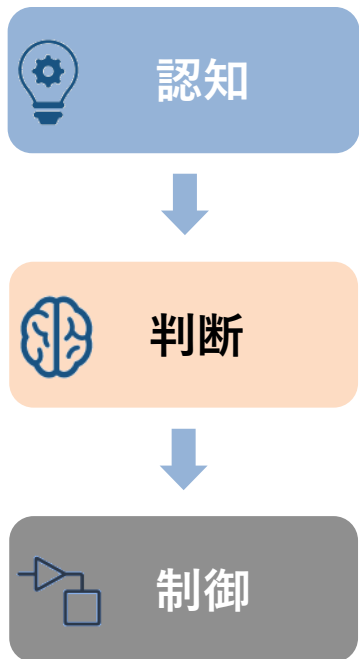


## 関連アルゴリズム・タスク代表例

- 画像処理
- 点群データ処理
- 自己位置推定、SLAM
- センサーフュージョン・トラッキング
- Deep Learning
- レーダー信号処理
- 組み込み機器への実装



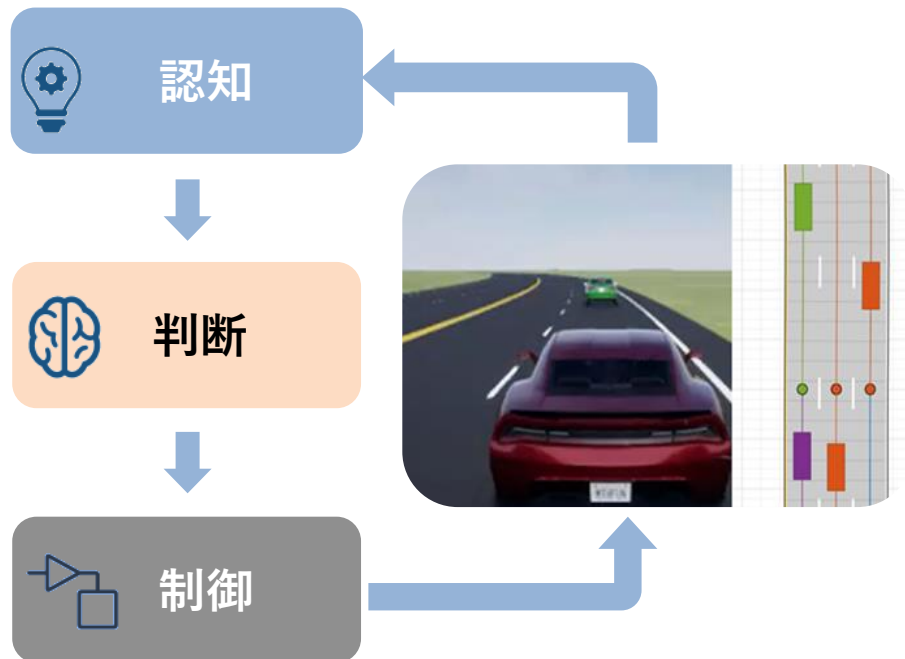
# ADAS・自動運転システムの開発 – 制御 with MATLAB and Simulink



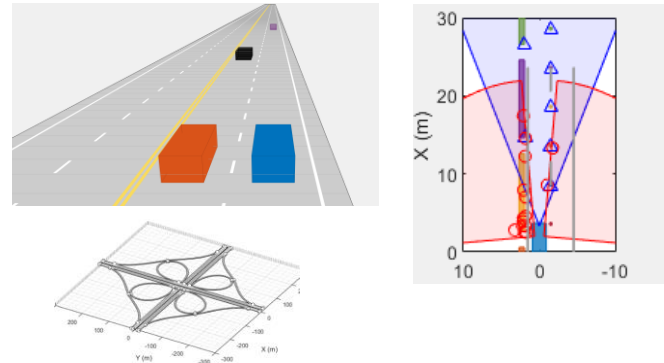
## 関連アルゴリズム・タスク代表例

- ログデータ、CANバス接続
- 強化学習
- 車両ダイナミクス
- レグレッションテスト
- ハードウェアを利用したプロトタイピング
- パス追従制御
- モデル予測制御
- 量産コード生成、実装
- AUTOSAR準拠
- ISO26262

# ADAS・自動運転システムの開発 – テストシナリオ・3D環境 with MATLAB and Simulink



## Driving Scenario Designer



抽象度の高い、簡易的表現  
確率的センサーモデル

- 統計的なセンサー特性に基づいた簡易シミュレーション
- 高速な実行時間

## Vehicle Dynamics Unrealエンジン連携

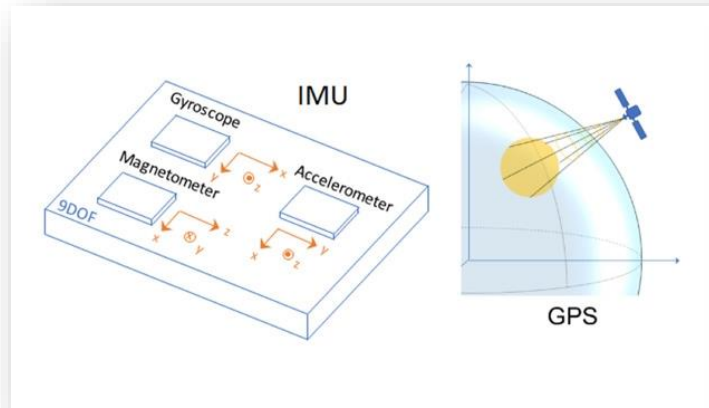


より“リアル”な環境、  
物理センサーモデル

- 認識系アルゴリズムを含む、より現実に近いシミュレーション

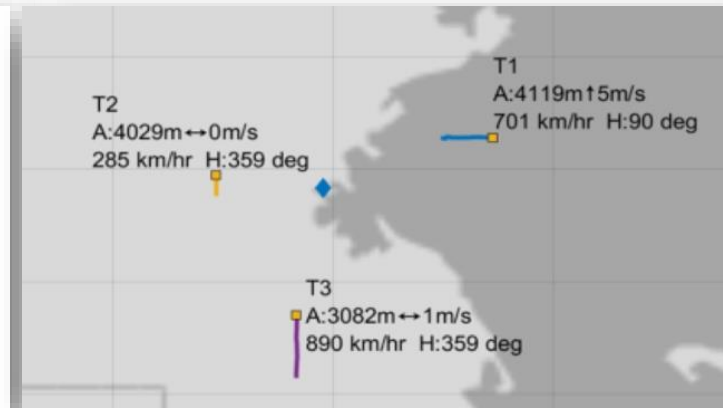
# センサーフュージョン&トラッキングとは? (1/2)

## 自己認識



Accelerometer, Magnetometer,  
Gyro, GPS...

## 周辺認識



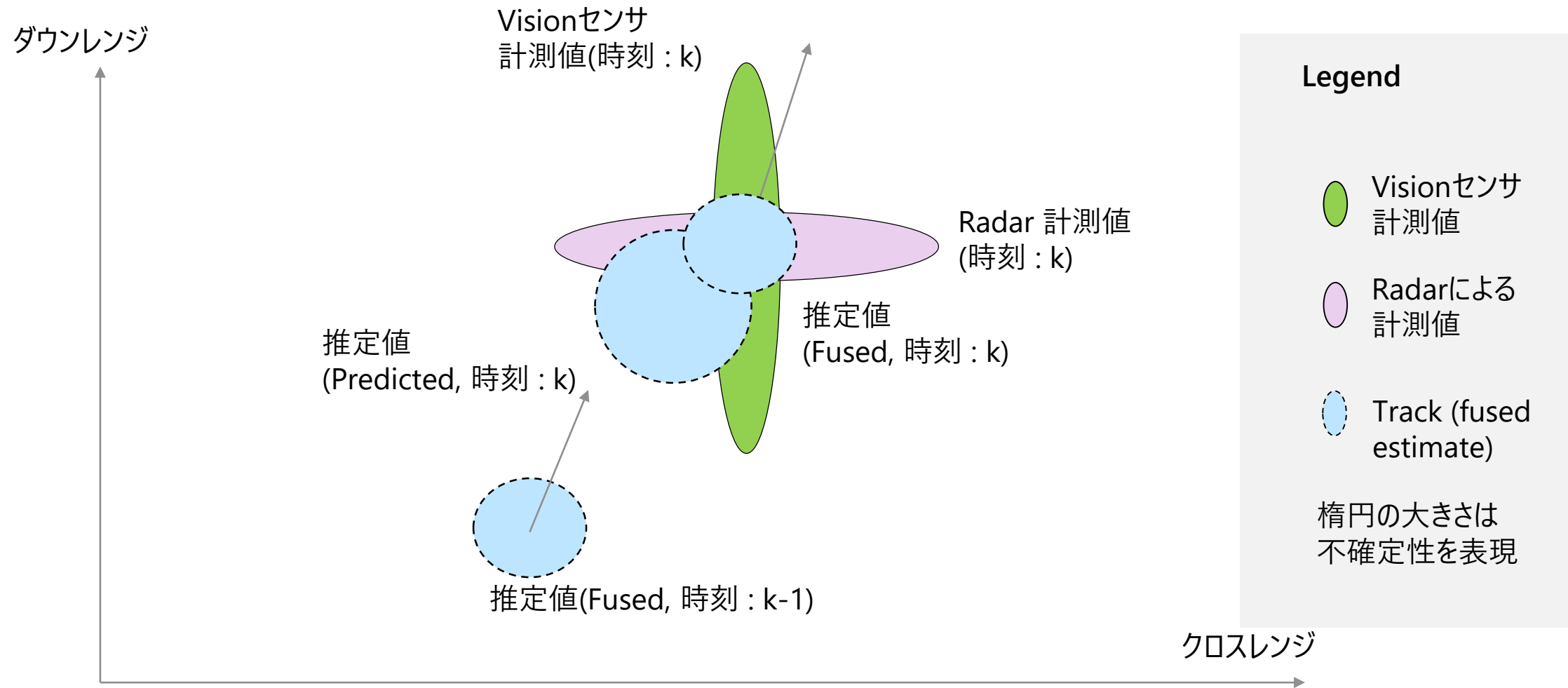
Radar, Camera, IR, Sonar, Lidar, ...

信号処理・  
画像処理等

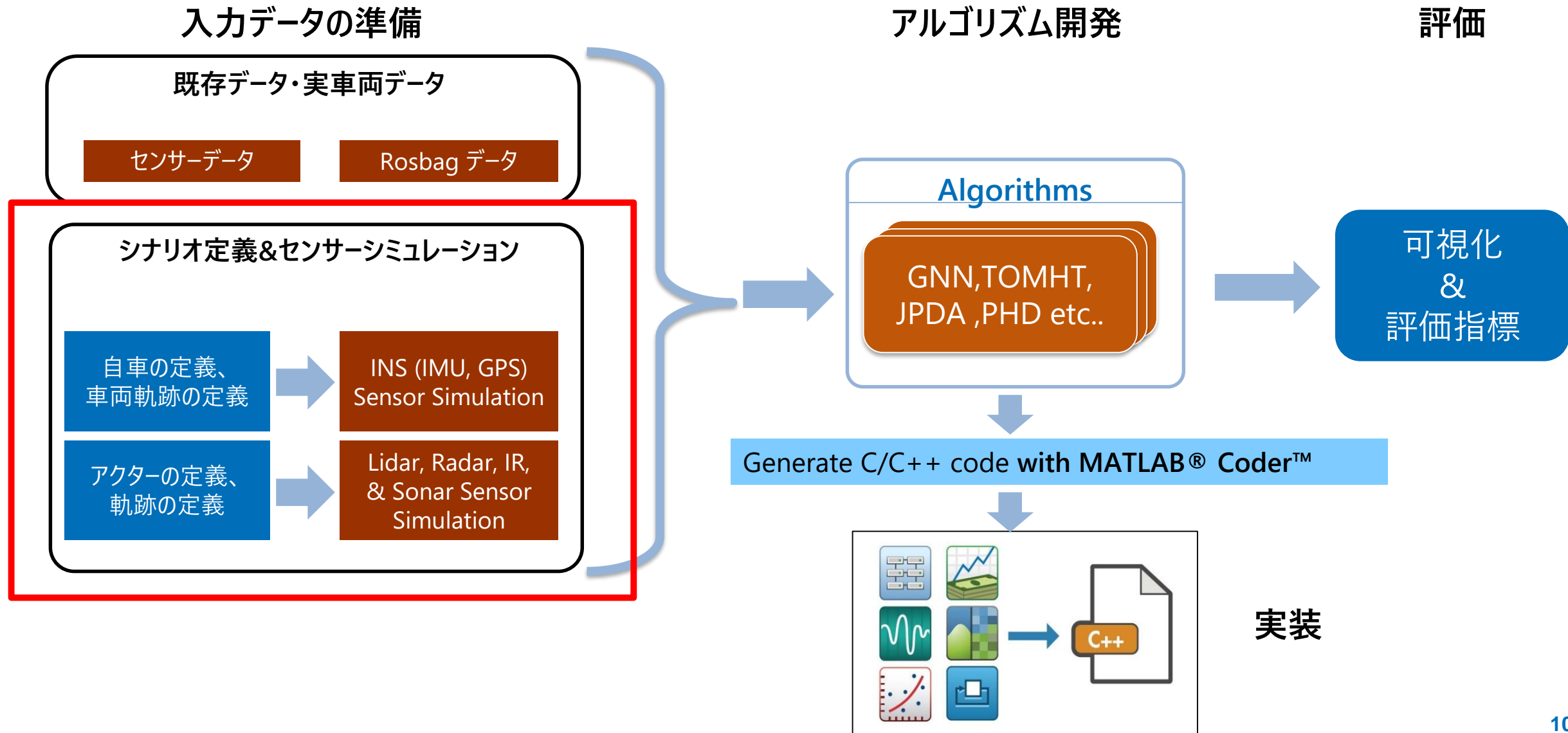
センサーフュージョン・  
トラッキング

制御

## センサーフュージョン&トラッキングとは? (2/2)



# センサーフュージョン&トラッキング アルゴリズム開発ワークフロー



# Driving Scenario Designer

高速動作、低コスト、制御系エンジニアの方でも使いやすいAPI

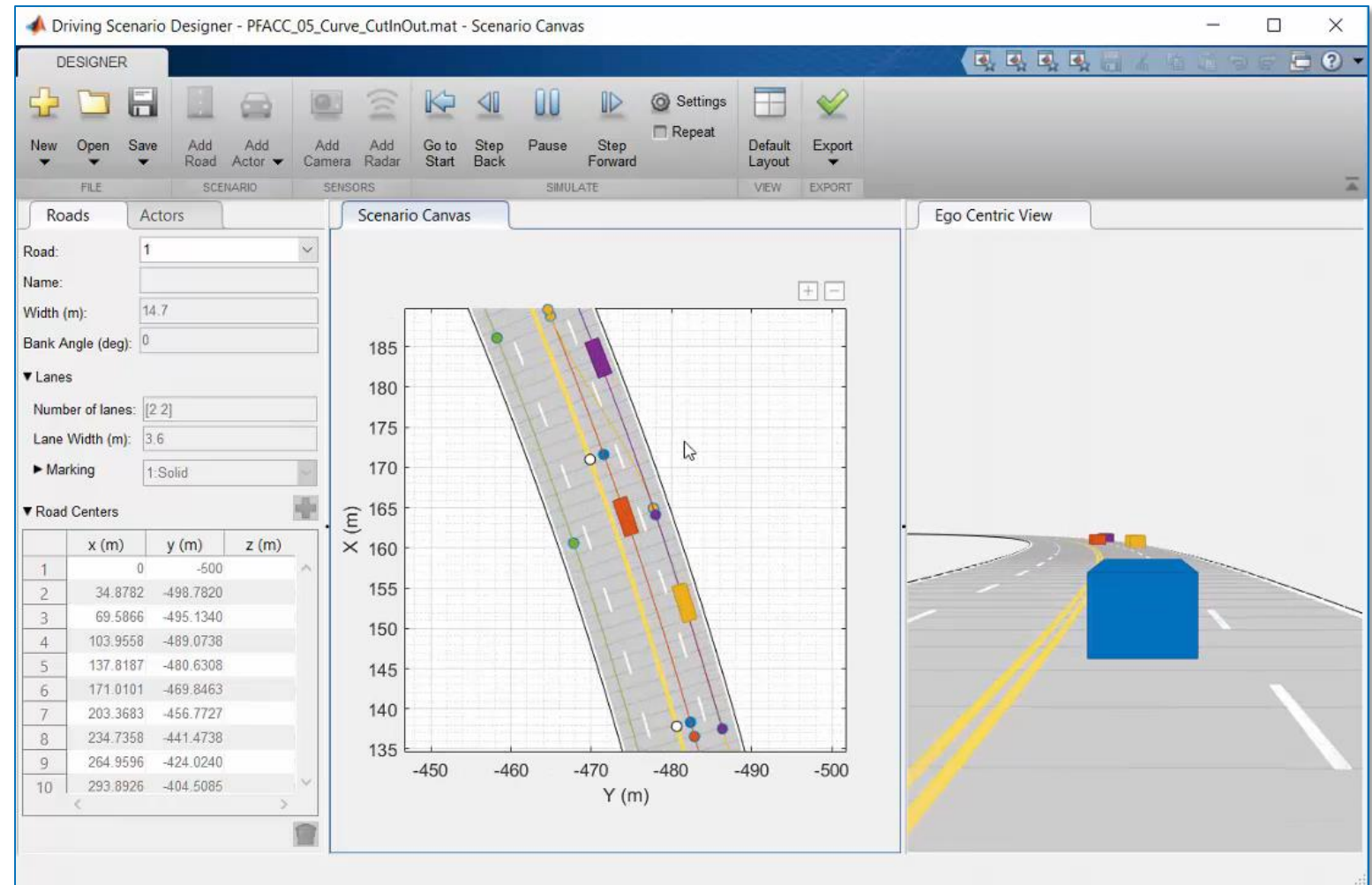
- 道路や車線、白線の作成
- 任意のアクター追加、軌道の定義
- アクターのサイズやRCS(Radar cross-section) の設定
- 事前定義済シナリオの活用
- OpenDRIVEやHERE HD Live Mapから道路情報をインポート
- 作成したシナリオはMATLABコードもしくはSimulinkモデルとして出力可能

[Driving Scenario Designer](#)

Automated Driving Toolbox™

Updated

R2020a



# Driving Scenario Designerの活用



ターゲット・プラットフォームの  
定義



オブジェクトの軌跡を  
定義



センサーの  
定義



シミュレーション



MATLABスクリプト/Simulink  
モデルの生成

車、トラック、歩行者等  
事前定義済のプラットフォーム  
(リストにないものはカスタマイズ可)

**VEHICLES**

- Car**  
Add a car to the scenario
- Truck**  
Add a truck to the scenario
- Building**  
Add a new building to the scenario

**OTHER**

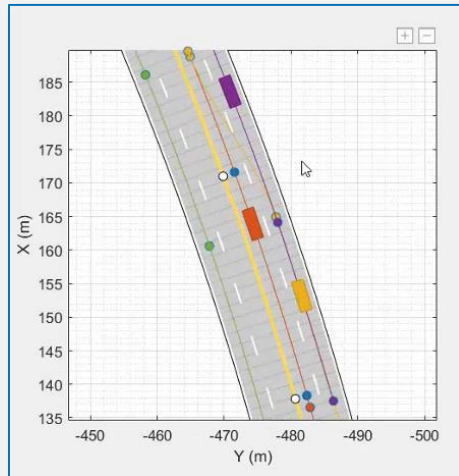
- Bicycle**  
Add a bicycle to the scenario
- Pedestrian**  
Add a pedestrian to the scenario
- Barrier**  
Add a barrier to the scenario
- マンホール**  
Add a new マンホール to the scenario

**CLASS SETTINGS**

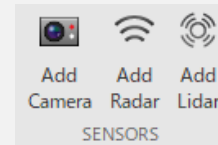
- New Actor Class**  
Create a new actor class
- Edit Actor Classes**  
Edit the actor class definitions

経路情報を設定:

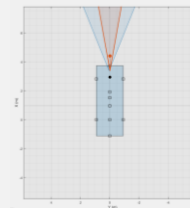
- 位置(x,y,z)
- 向き(Yaw)
- 速度 など



センサーのコンフィグレーション設定



センサーの取り付け



センサーパラメータの設定

**Sensor Placement**

X (m): 3.4 Y (m): 0.01 Height (m): 1.1

Roll: 0 Pitch: 1 Yaw: 0

**Camera Settings**

Focal Length X: 800 Y: 800

Image Width: 640 Height: 480

Principal Point X: 320 Y: 240

**Detection Parameters**

Detection Type: Objects

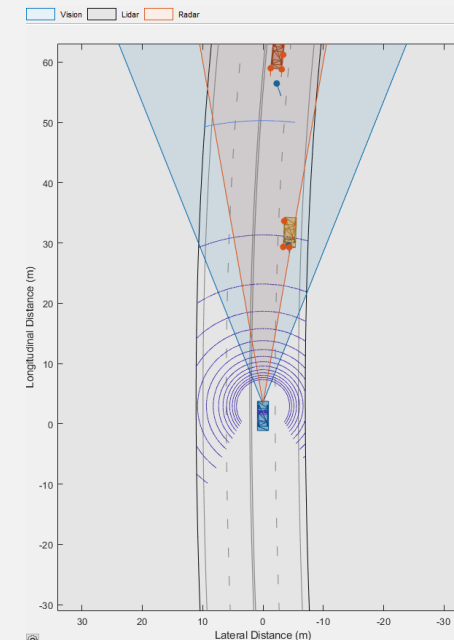
Detection Probability: 0.9

False Positives Per Image: 0.1

☐ Limit # of Detections:

Detection Coordinates: Ego Cartesian

センサーデータの生成



MATLAB

**MATLAB Function**  
Generate MATLAB function for the driving scenario and sensors

**Export Sensor Data**  
Export sensor data from last simulation run to base workspace

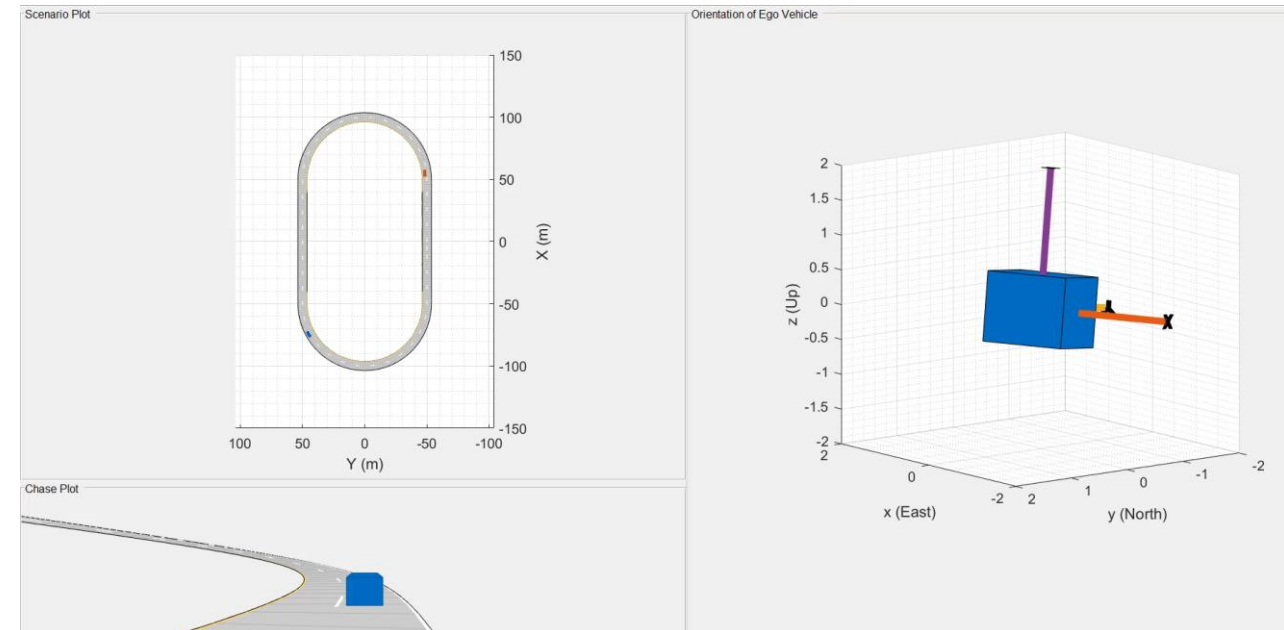
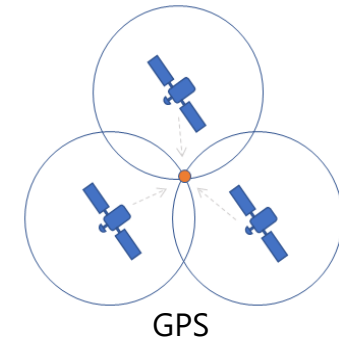
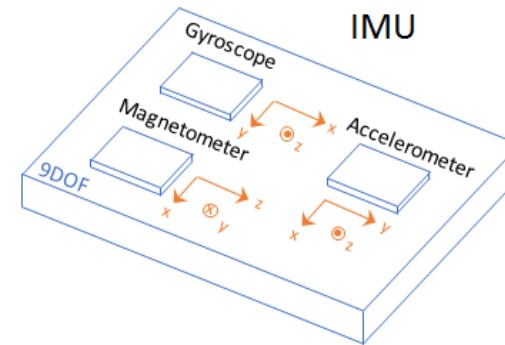
SIMULINK

**Simulink Model**  
Generate Simulink model for the driving scenario and sensors

**Export Sensor Simulink Model**  
Generate Simulink model for only your sensors

# 利用可能な様々なセンサモデル

センサモデル名	関数・オブジェクト名
IMU	imuSensor
GPS	gpsSensor
Altimeter	altimeterSensor
INS/GPS	insSensor
Active Radar	monostaticRadarSensor
Passive Radar	radarSensor
Jammer/RF Emitter	radarEmitter
Bistatic radar	radarSensor+radarEmitter
Active Sonar	sonarEmitter+sonarSensor
Passive Sonar	sonarSensor
Jammer/Sound Emitter	sonarEmitter
Bistatic sonar	sonarEmitter+sonarSensor
EO/IR	irSensor

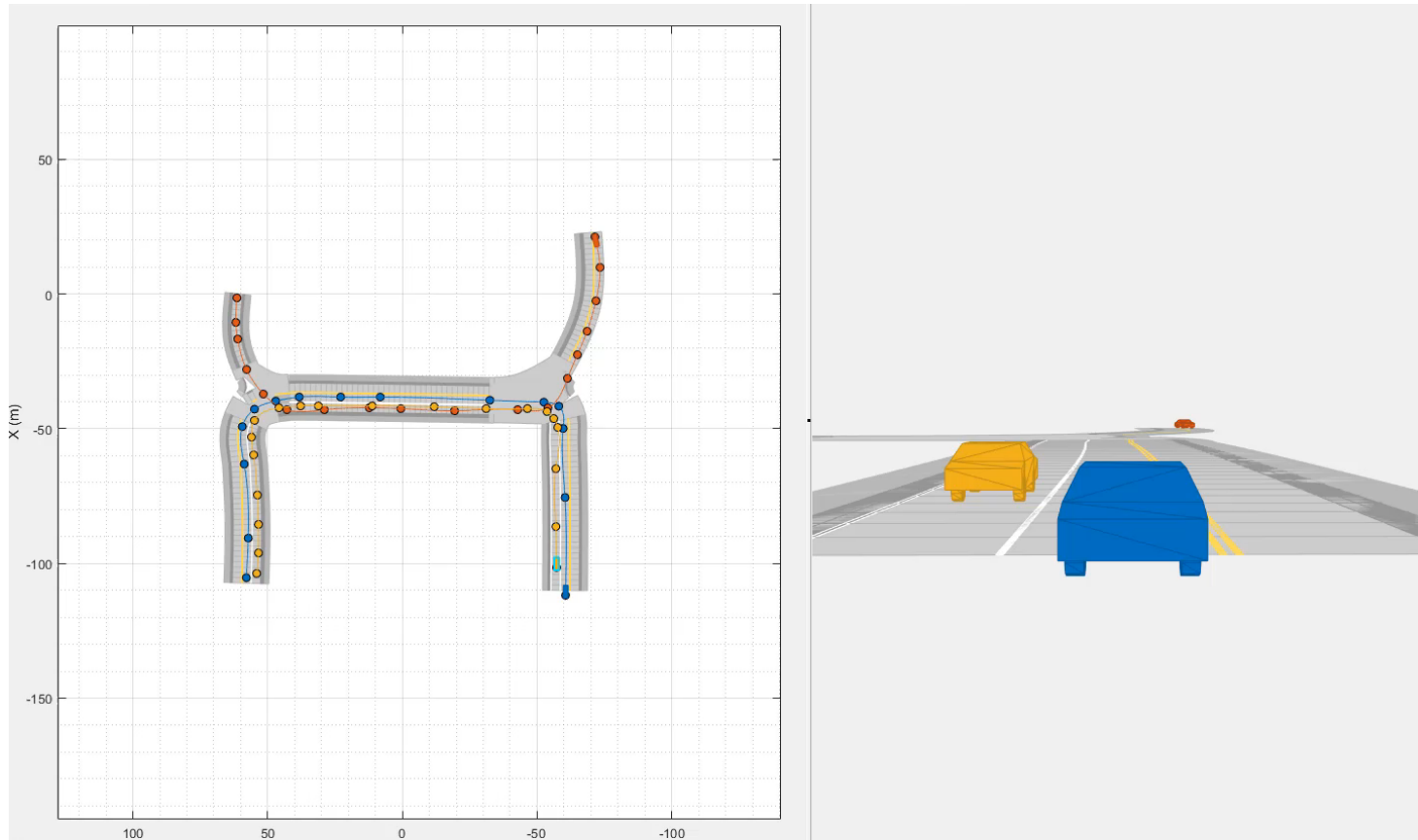


# センサ出力フォーマットと各プロパティの概要

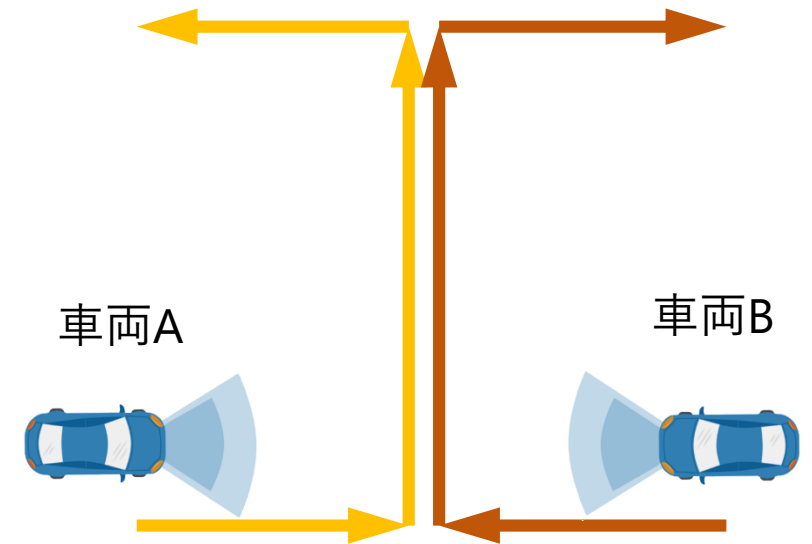
## objectDetection

プロパティ	概要	Sensor Fusion & Trackingアルゴリズムにおける重要性
Time	計測時刻	ある時刻におけるTrackの予測、その結果を元にした距離の算出に必要
SensorIndex	各センサーに付与されたインデックス	センサーの種類を区別し、1センサーあたり1Detectionになるようにするために必要
Measurement	計測結果	フィルタ(動作モデル)の評価・更新に必要
MeasurementNoise	計測結果の確からしさを示す共分散行列	フィルタ(動作モデル)の評価・更新に必要
MeasurementParameters (Optional)	センサーの姿勢、位置情報等のパラメータ	非線形フィルタ(動作モデル)の評価・更新に必要な場合がある
ObjectClassID (Optional)	物体種類に紐づけられたインデックス。 (歩行者、トラック等)	既存のTrackに対し、ことなる物体クラスのDetectionを割り当てないようにすることができる。
ObjectAttributes (Optional)	ターゲット毎の固有インデックス	上記Class IDと組み合わせると真値が得られる。

# 実車では再現の難しいテストシナリオの作成

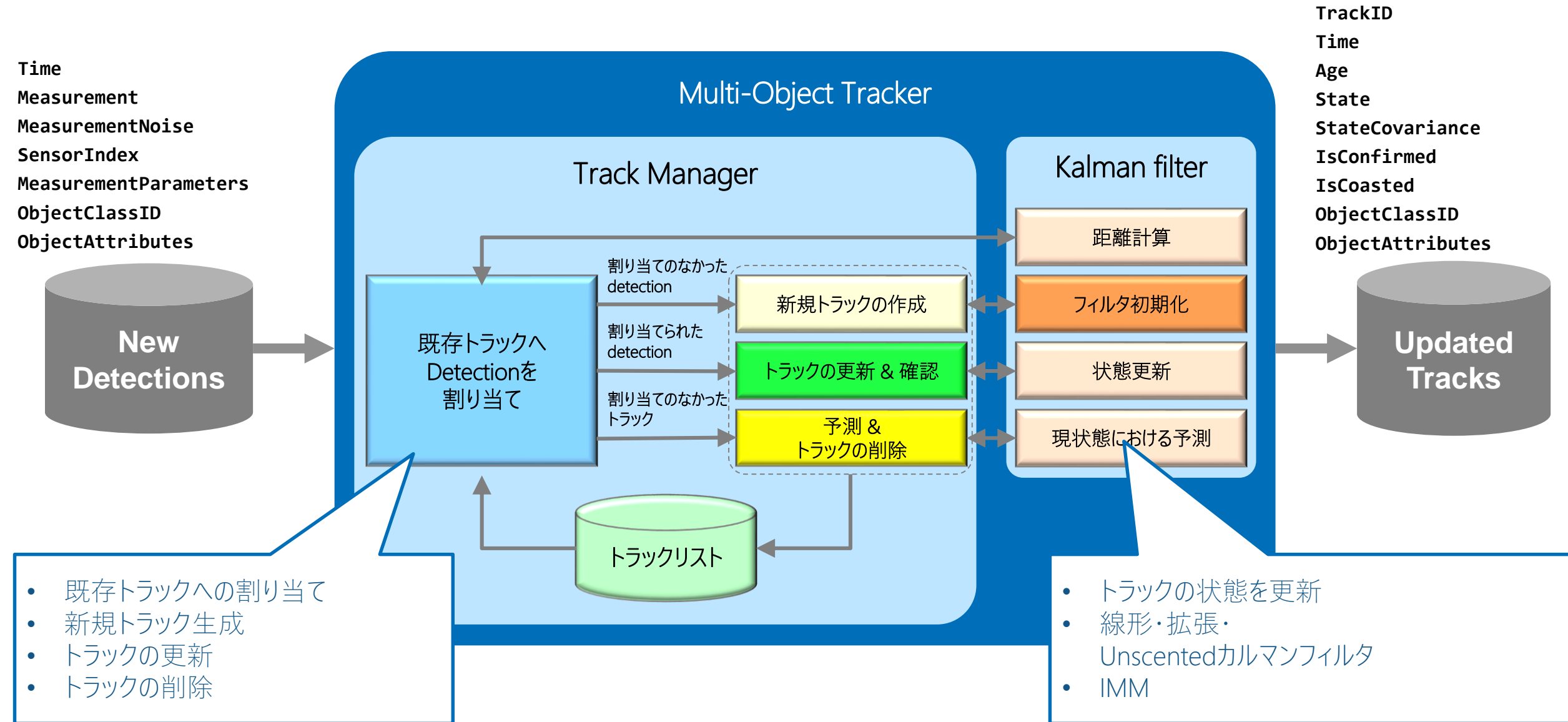


車両の動き



車両が交差したところでトラックを正しく維持できるか？

# オブジェクトトラッカーの構成例

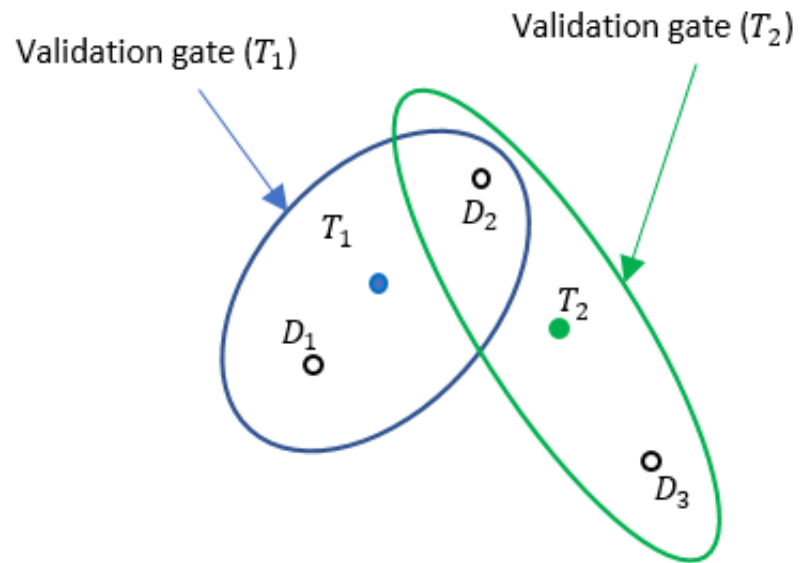


## Muiti Object Trackerの種類と概要

### Sensor Fusion and Tracking Toolbox™

Assignment Name	Description	Function or Class
Global Nearest Neighbor (GNN)	単一の仮説に基づく割り当て	trackerGNN assignmunkres assignjv assignauction
Joint Probabilistic Data Association (JPDA)	全ての既存トラックを考慮して各Detectionの尤度(各トラックに割り当てられる可能性)が計算される	trackerJPDA jpdaEvents
Track-Oriented Multiple-Hypothesis Tracking (TOMHT)	各トラックに対して複数の仮説(割り当てられる可能性のある全てのDetectionを考慮、もしくは割り当てが無い可能性)を立て、次の更新にも活用する	trackerTOMHT assignTOMHT trackBranchHistory clusterTrackBranches compatibleTrackBranches pruneTrackBranches
Hypothesis-Oriented Multiple-Hypothesis Tracking (HOMHT)	K-best問題を解くことによって割り当てを決定し、トラックを更新する	Assignkbest
Probability Hypothesis Density (PHD)	明確な割り当ては行わない。トラッキングの問題を未知のランダムなオブジェクトの集合としてモデル化し、Detectionに基づいて各ロケーションにおける確率を推定する	trackerPHD ggiwphd partitionDetections

# JPDA(Joint Probabilistic Data Association)



2D or 3D楕円の空間境界。  
既存Trackの確率情報(状態推定や共分散等)により定義

Validation Matrix

$$\Omega = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

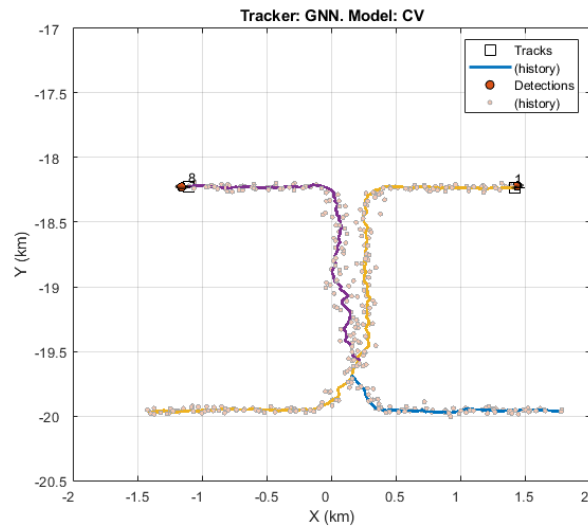
$\swarrow$   $\uparrow$   $\swarrow$   
 T0    T1    T2

特定の仮説に基づいて、FJE(Feasible Joint Event)を生成

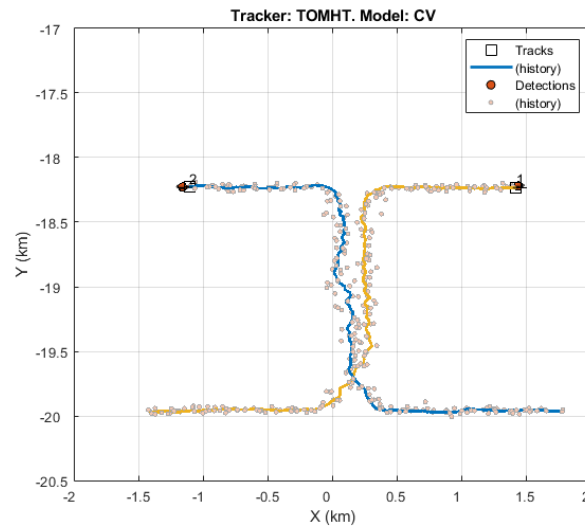
$$\begin{aligned} \Omega_1 &= \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \Omega_2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \Omega_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \Omega_4 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \\ \Omega_5 &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \quad \Omega_6 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \Omega_7 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \Omega_8 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

各JFEに対して、事後確率を計算

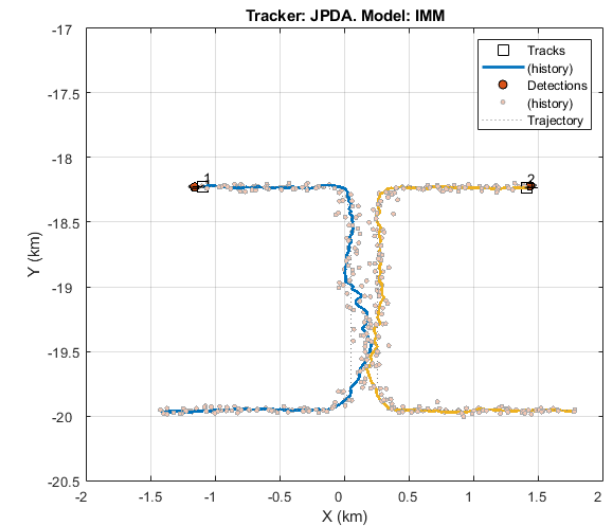
# 各種トラッカー・動作モデルの定義例



```
tracker = trackerGNN( ...
    'FilterInitializationFcn',@initCVFilter,...
    'MaxNumTracks', numTracks, ...
    'MaxNumSensors', 1, ...
    'AssignmentThreshold',gate, ...
    'TrackLogic', 'Score', ...
    'DetectionProbability', pd, ...
    'FalseAlarmRate', far, ...
    'Volume', vol, 'Beta', beta);
```



```
tracker = trackerTOMHT( ...
    'FilterInitializationFcn',@initCVFilter, ...
    'MaxNumTracks', numTracks, ...
    'MaxNumSensors', 1, ...
    'AssignmentThreshold', [0.2, 1, 1]*gate,...
    'DetectionProbability', pd, ...
    'FalseAlarmRate', far, ...
    'Volume', vol, 'Beta', beta, ...
    'MaxNumHistoryScans', 10,...
    'MaxNumTrackBranches', 5,...
    'NScanPruning', 'Hypothesis', ...
    'OutputRepresentation', 'Tracks');
```



```
tracker = trackerJPDA( ...
    'FilterInitializationFcn',@initIMMFilter,...
    'MaxNumTracks', numTracks, ...
    'MaxNumSensors', 1, ...
    'AssignmentThreshold',gate, ...
    'TrackLogic','Integrated',...
    'DetectionProbability', pd, ...
    'ClutterDensity', far/vol, ...
    'NewTargetDensity', beta,...
    'TimeTolerance',0.05);
```

# トラッキングパフォーマンスの解析

## ■ trackAssignmentMetrics

	1	2	3	4	5	6	7	8	9	10	11	12	
	TrackID	AssignedTruthID	Surviving	TotalLength	DeletionStatus	DeletionLength	DivergenceStatus	DivergenceCount	DivergenceLength	RedundancyStatus	RedundancyCount	RedundancyLength	FalseT
1	1	3	1	190	0	0	0	25	83	0	1	13	
2	2	NaN	0	73	0	0	1	12	32	0	0	0	
3	8	2	1	111	0	0	0	12	58	0	0	0	

### — トラックの割り当てに関する評価指標の提供

- Redundancy State : 別のトラックが割り当てられていた真値に対して新たなトラックが割り当てられる状態
- Divergence State : トラックが削除される閾値を超えてしまった状態

## ■ trackErrorMetrics

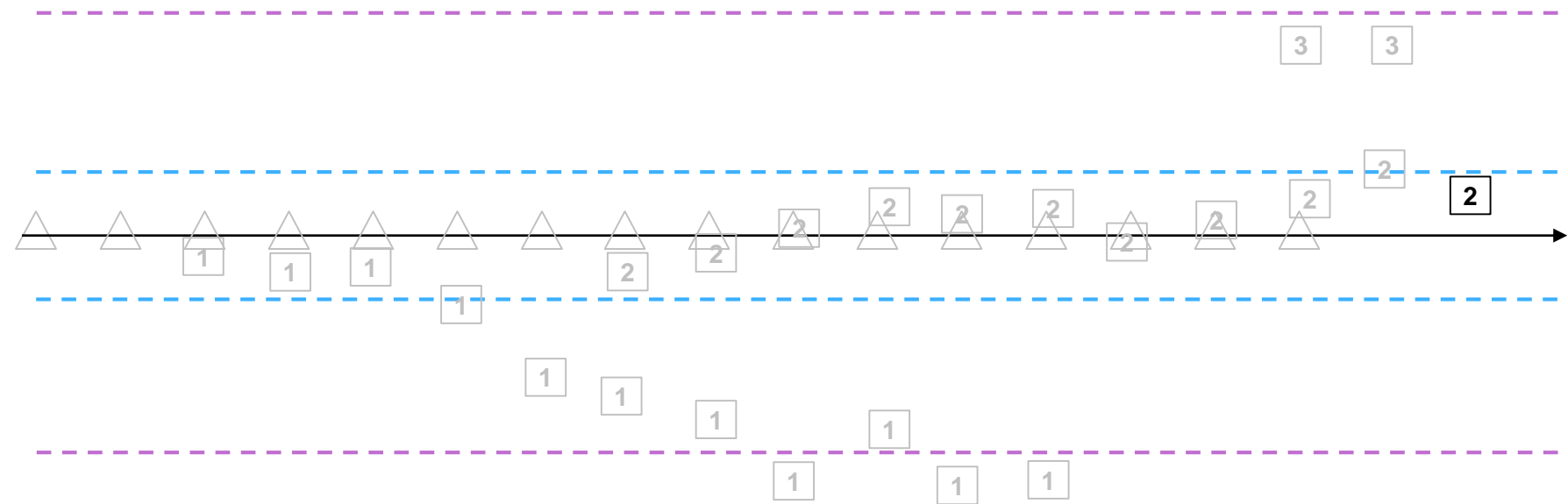
	1	2	3	4	5
	TrackID	posRMS	velRMS	posANEES	velANEES
1	1	59.4606	26.1189	7.9331	3.4950
2	2	49.1732	38.0756	5.3600	2.6999
3	8	62.9136	28.8768	7.2004	3.0465

- 位置、速度、加速度、姿勢の比較
- RMSE, ANEES(Average Normalized-Estimation Error Squared)

>> trackerGNN, CV



# trackAssignmentMetricsについて



**Legend**

1

Track

△

Truth

---

Assignment Distance

---

Divergence Distance

Track ID	Surviving	Total Length	Deletion		Divergence			Redundancy			False Track		Swap Count
			Status	Length	Status	Count	Length	Status	Count	Length	Status	Length	
1	false	11	false	0	true	2	3	false	0	0	false	0	0
2	true	11	true	2	false	0	0	false	1	6	false	0	0
3	false	2	false	0	false	0	0	false	0	0	true	2	0

# OSPA(Optimal sub-pattern assignment) Metric

- trackAssignmentMetricsとtrackErrorMetricsを補完
- トラッキングパフォーマンス(MoEs)をシンプルに表現

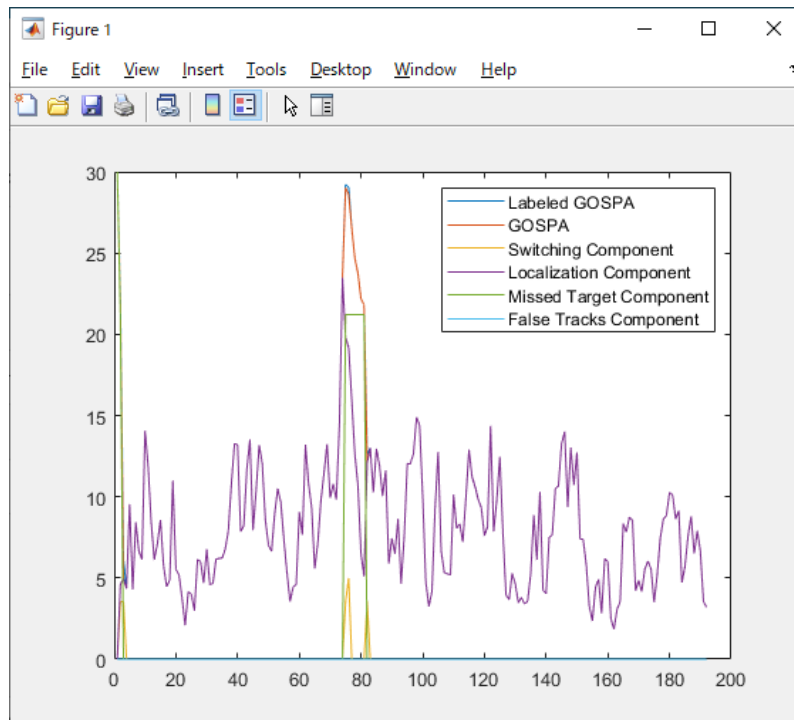
**trackOSPA****Metric** **R2019b**

Optimal subpattern assignment (OSPA) metric

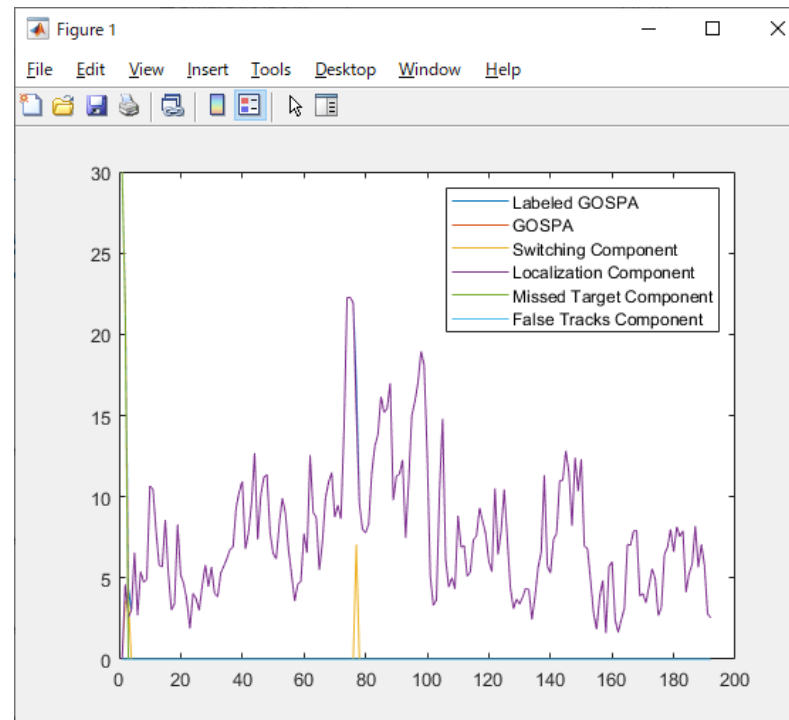
**trackGOSPAMetric** **R2020a**

Generalized optimal subpattern assignment (GOSPA) metric

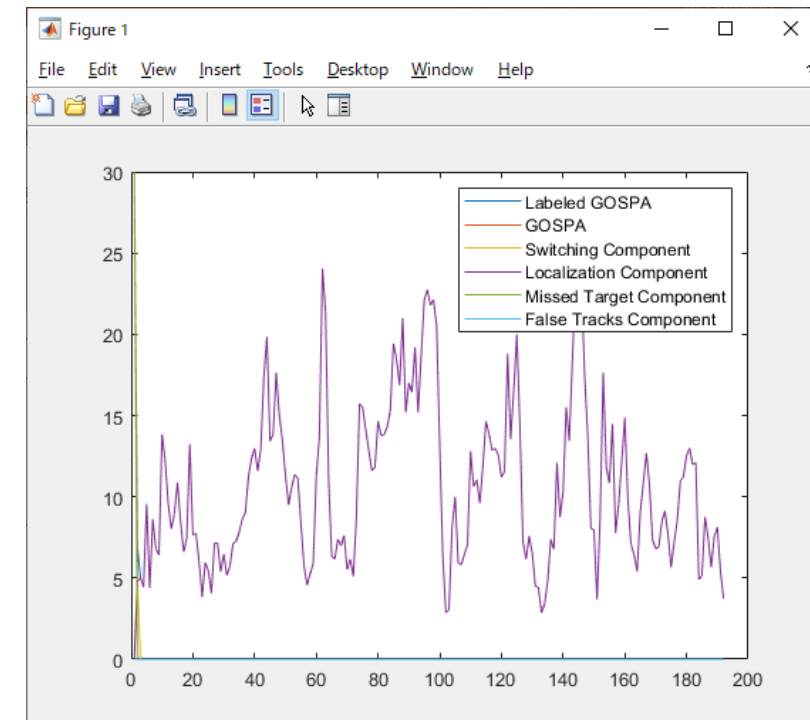
>> trackerGNN, CV



>> trackerTOMHT, CV

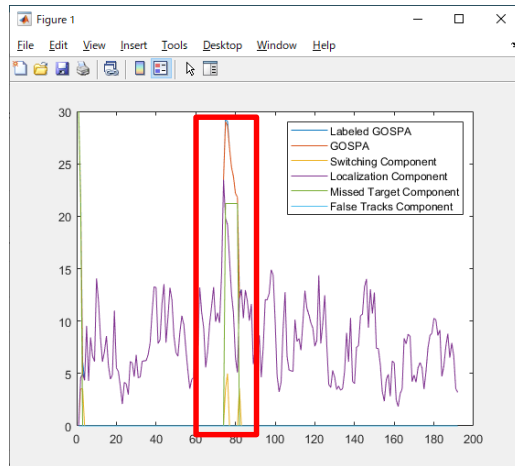


>> trackerJPDA, IMM

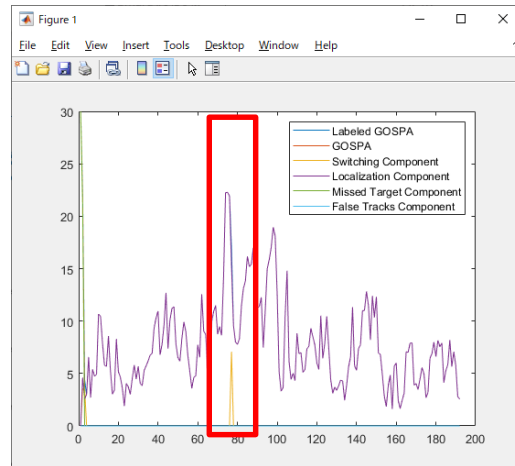


# パフォーマンスを比較して最終的にアルゴリズムを決定

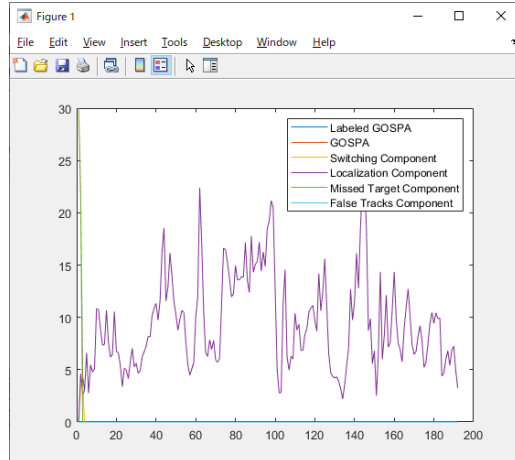
>> trackerGNN, CV



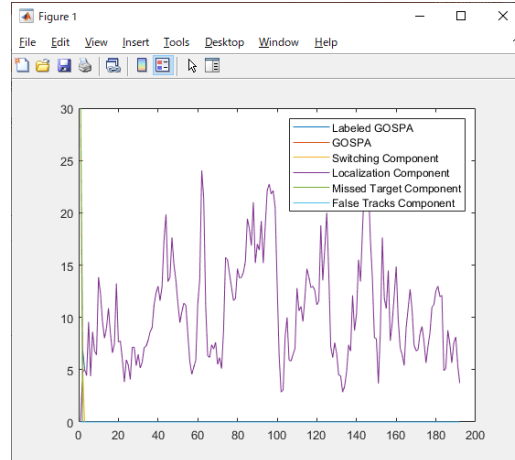
>> trackerTOMHT, CV



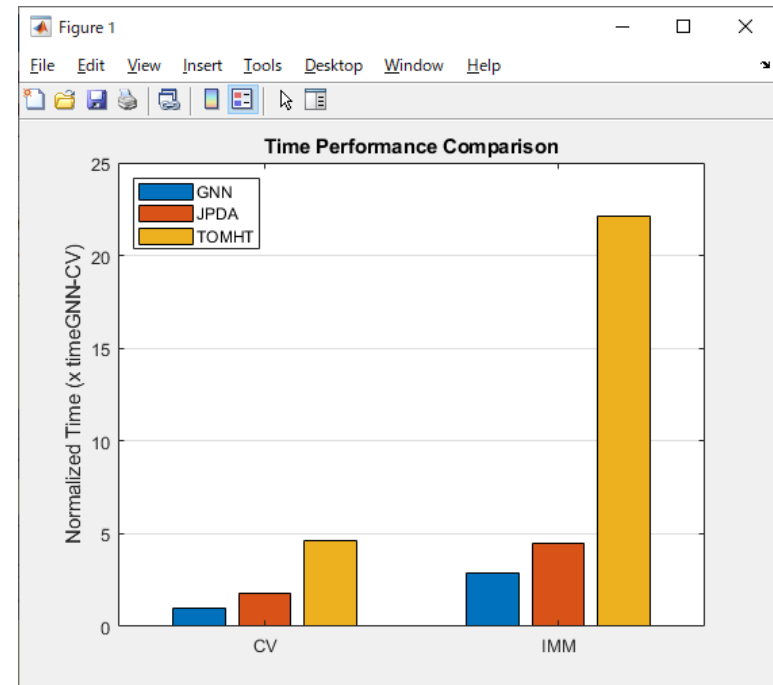
>> trackerTOMHT, IMM



>> trackerJPDA, IMM



## 計算コスト

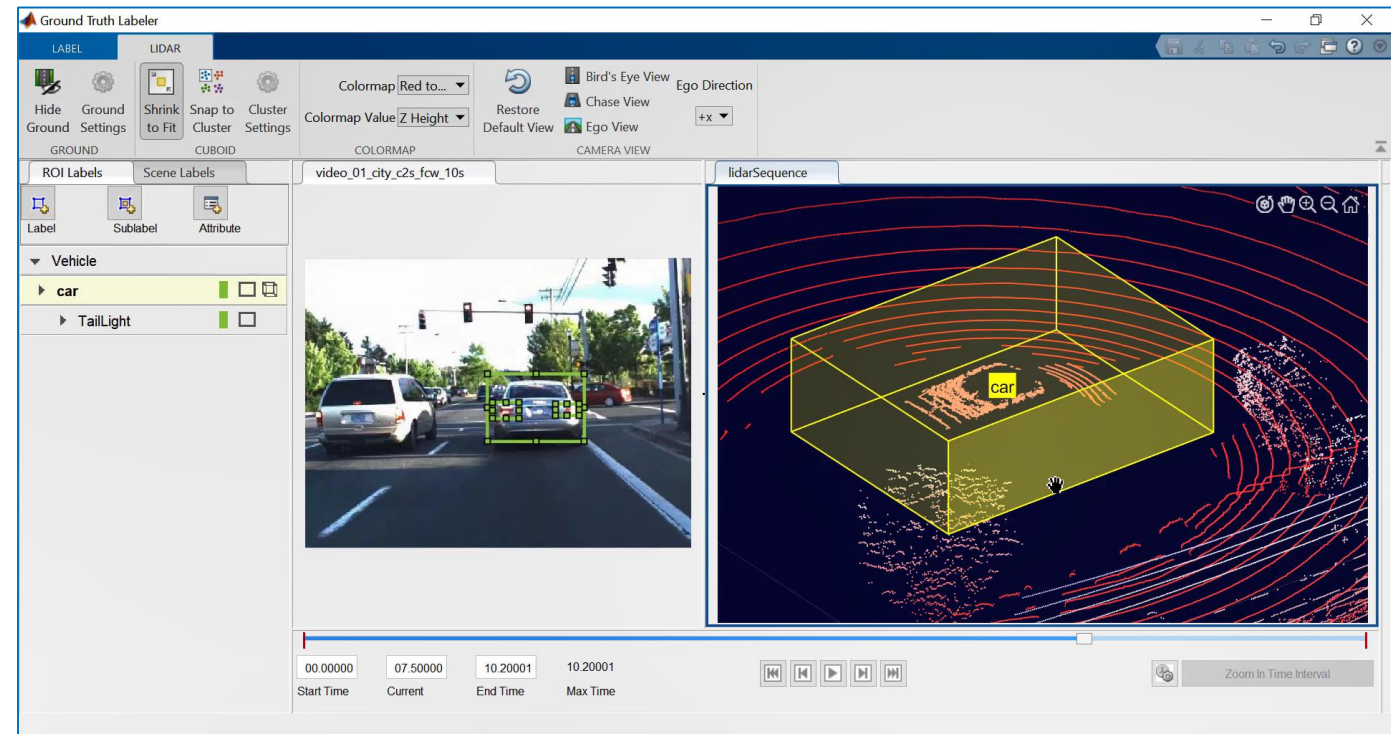


**trackerJPDA + IMM**

# Ground Truth Labeler App

点群データを含む、複数のセンサーデータに対するラベリング

- 各種センサーデータに対する対話型のラベリング
  - 矩形ラベル (ROI)
  - ポリライン ROI
  - ピクセル ROI (semantic segmentation)
  - Cuboid-直方体 (lidar)
  - シーン
- 物体検出やトラッキング等を利用した、ラベリング自動化アルゴリズム
- 自動化アルゴリズムのカスタマイズ&登録
- 可視化ウィンドウの配置・サイズのカスタマイズ
- ラベル情報の外部

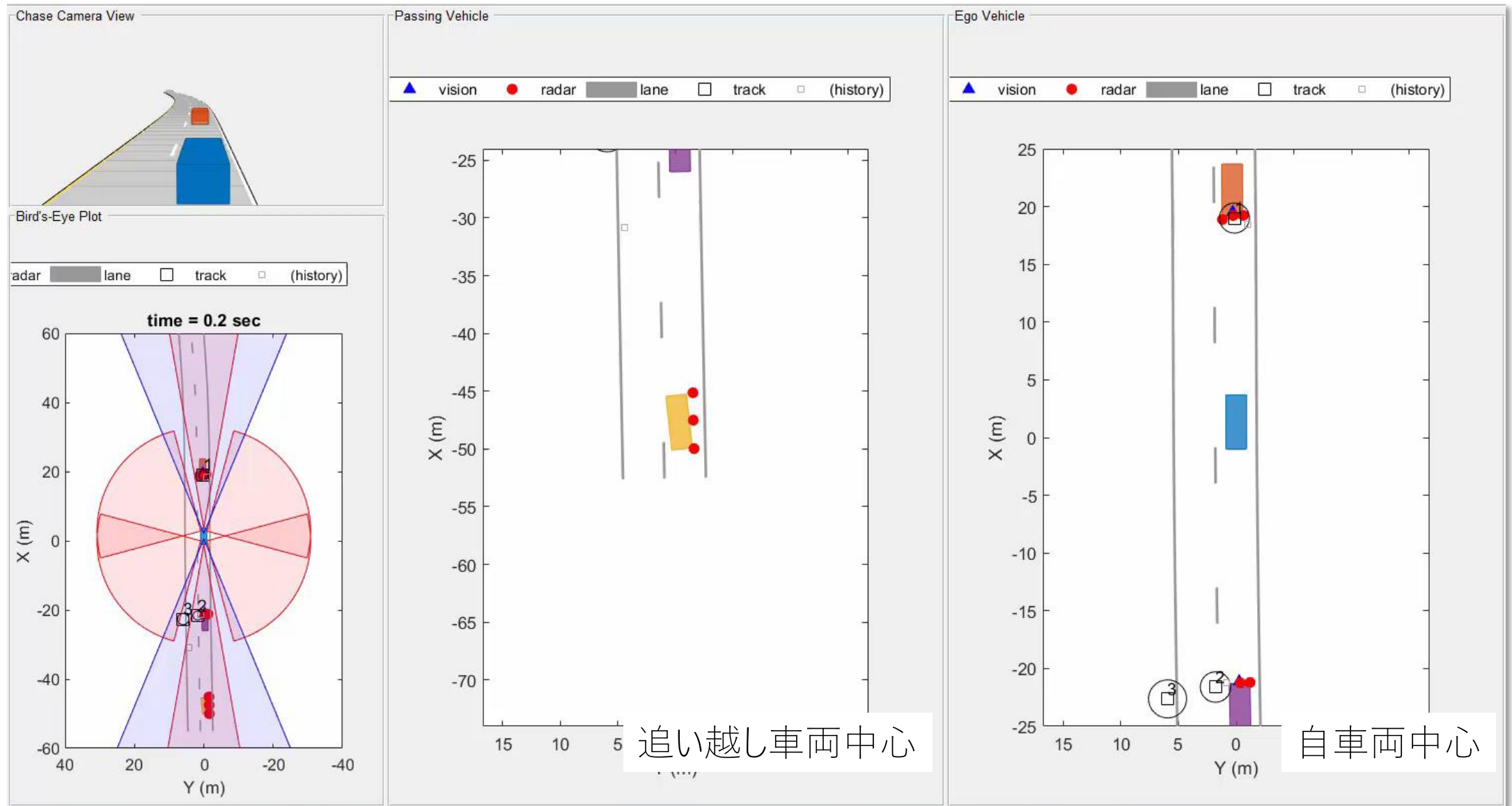


[Ground Truth Labeler](#)

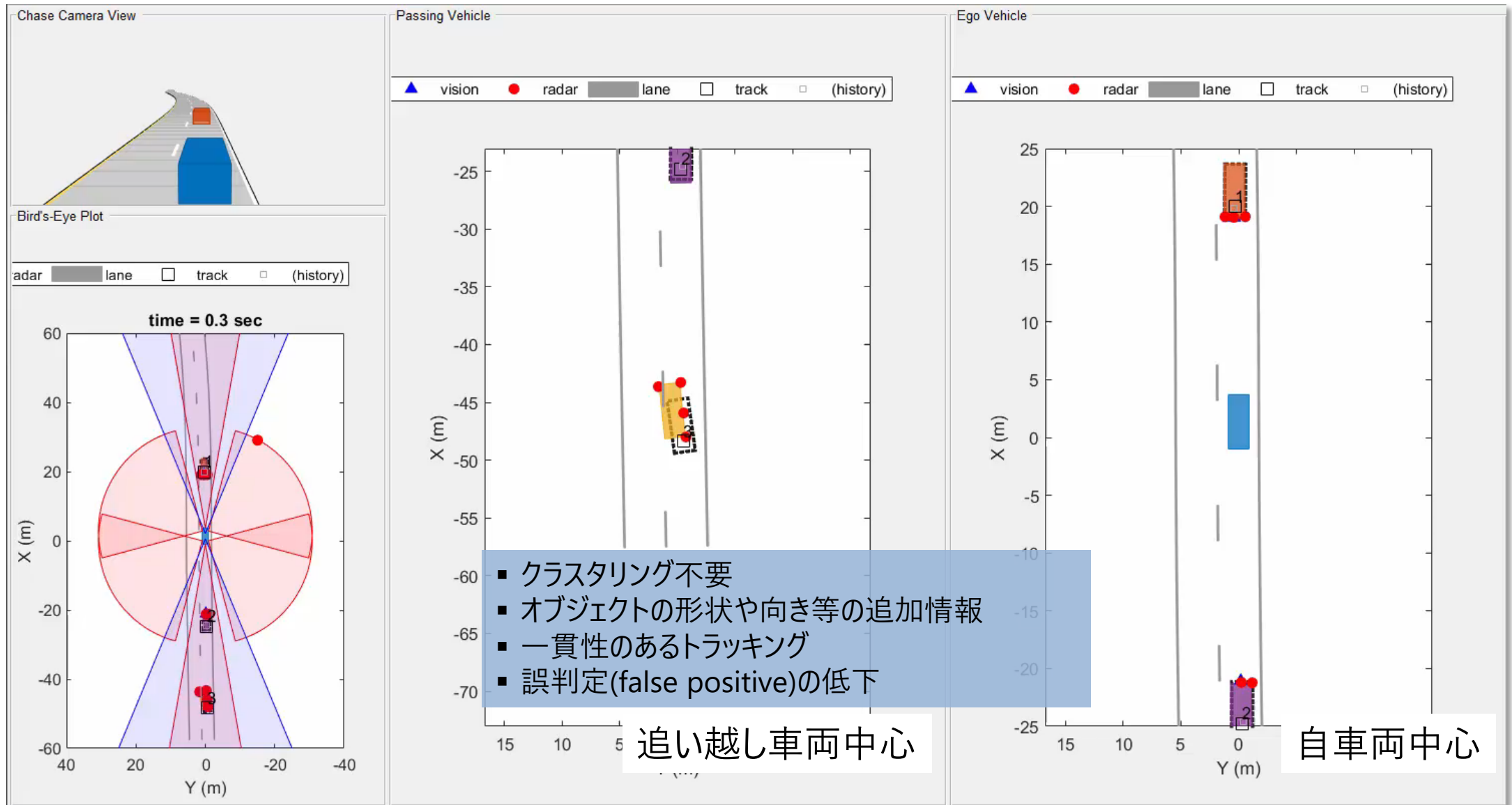
Automated Driving Toolbox™

Updated **R2020a**

# Point Object Tracker

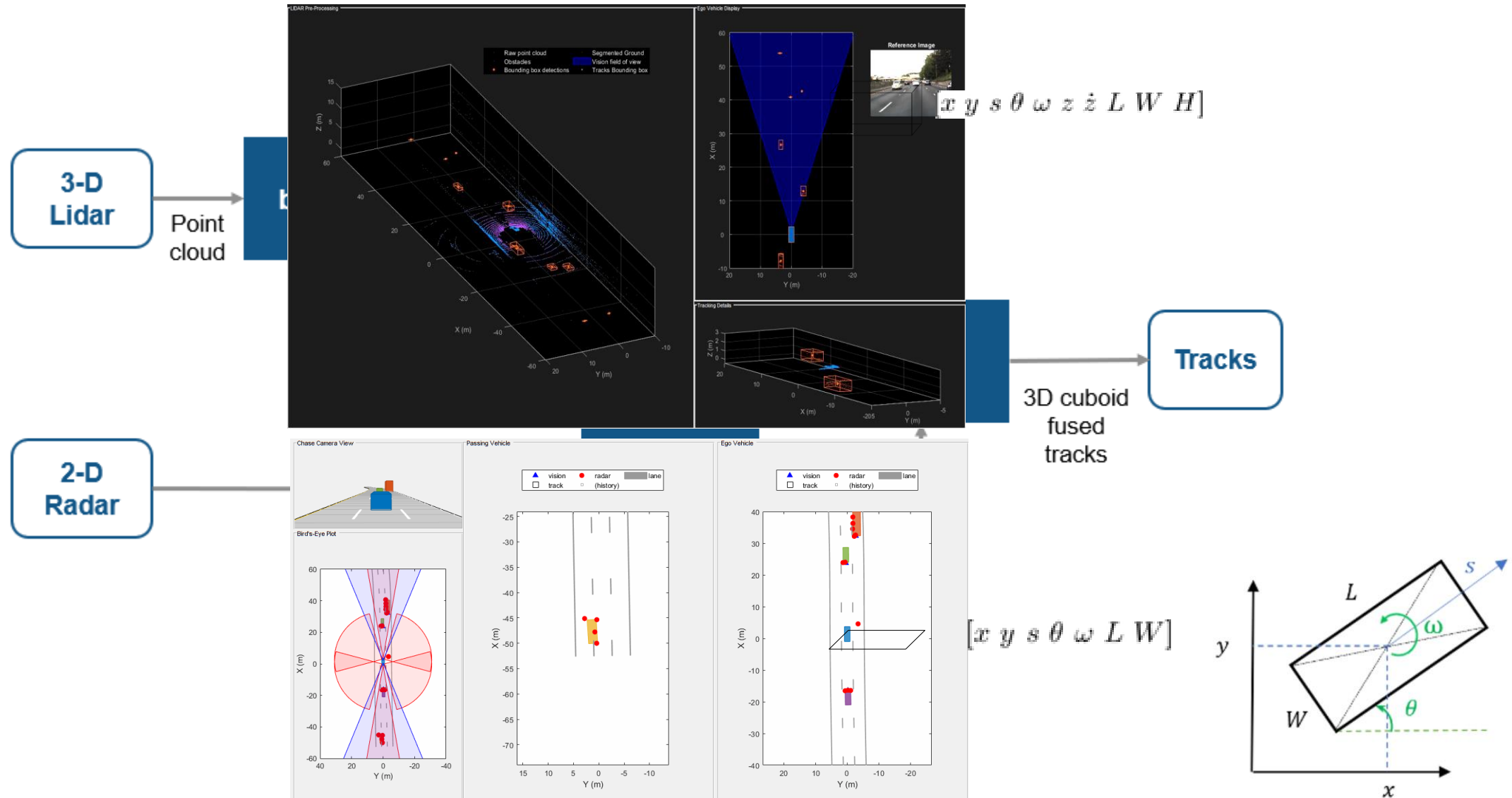


# Extended Object Tracker – 位置、車両姿勢、サイズのトラッキング



# 2D Radar + 3D LiDAR でトラッキング精度を改善

2つの異なるセンサー・モダリティ



# 2D Radar + 3D LiDAR Trackレベルフュージョン

## Sensor Fusion and Tracking Toolbox™

シナリオ  
定義

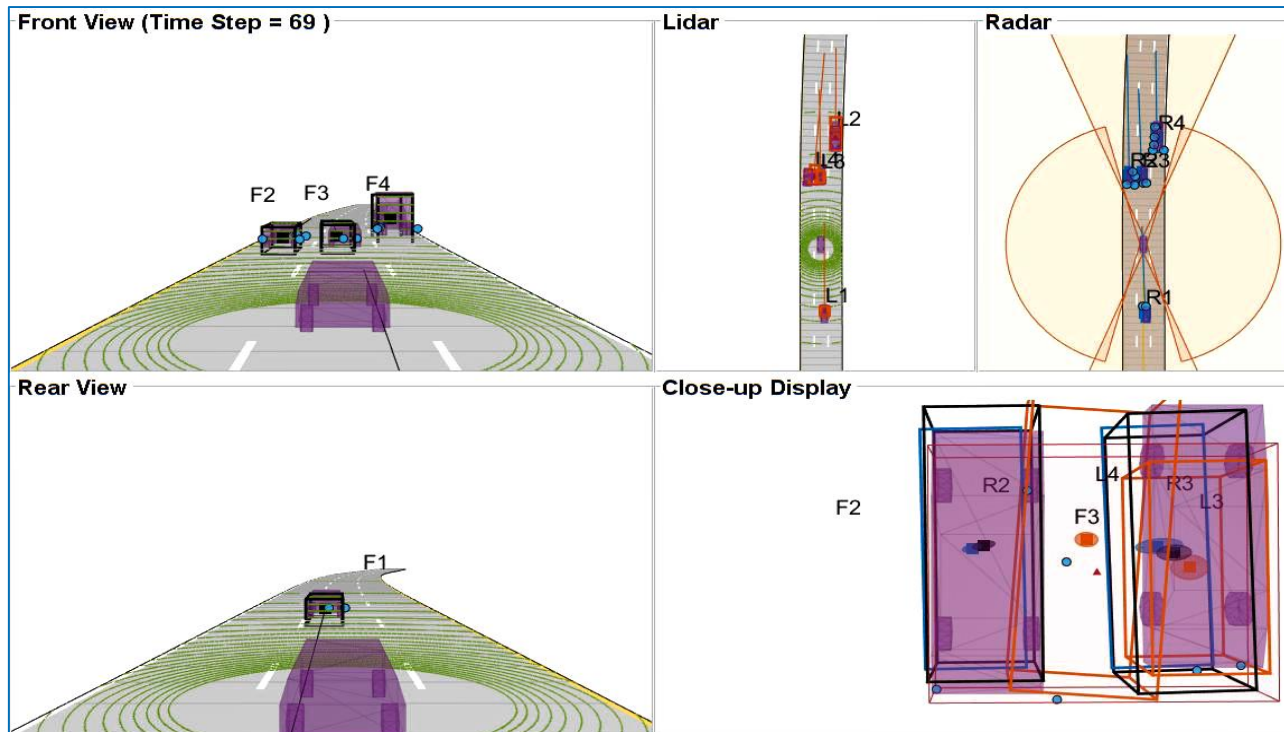
物体検出  
(LiDAR)

トラッキング  
(LiDAR)

トラッキング  
(Radar)

Track  
フュージョン

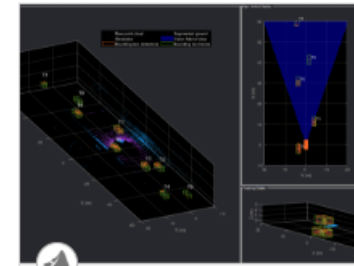
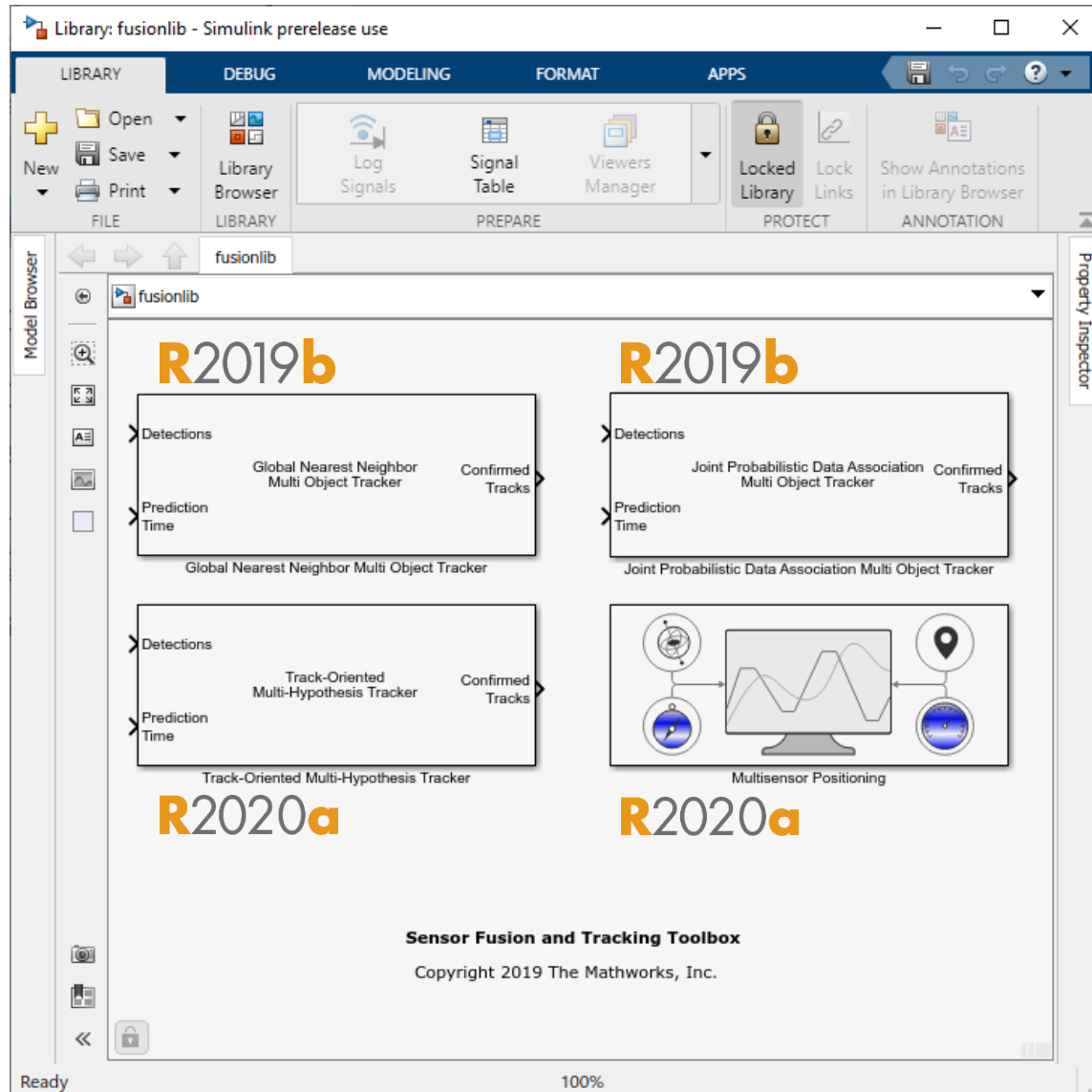
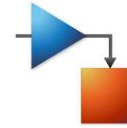
メトリクスの  
評価



- Driving Scenarioによるシナリオ定義
- 点群データ前処理、クラスタリング
- IMM-UKF(JPDAトラッカー)による点群クラスタのトラッキング
- GM-PHD(拡張オブジェクトトラッカー)によるレーダー検出のトラッキング
- Track-Levelフュージョン
- 各種メトリクスによるパフォーマンス評価

Track-Level Fusion of Radar and Lidar Data

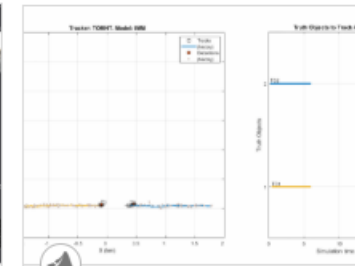
# SFTTで提供されるSimulinkブロック



## Track Vehicles Using Lidar Data in Simulink

Track vehicles using measurements from a lidar sensor mounted on top of an ego vehicle. Due to high resolution capabilities of the lidar

**R2019b** [Open Model](#)

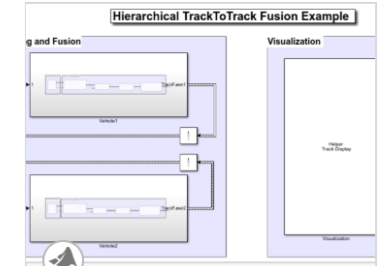


## Track Closely Spaced Targets Under Ambiguity in Simulink

Track objects in Simulink® with Sensor Fusion and Tracking Toolbox™ when the association of sensor detections to tracks is

**R2019b** [Open Model](#)

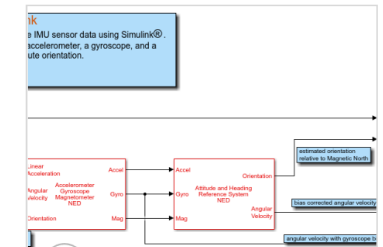
(Updated)



## Track-to-Track Fusion for Automotive Safety Applications in Simulink

Perform track-to-track fusion in Simulink® with Sensor Fusion and Tracking Toolbox™. In the context of autonomous driving, the example

**R2020a**



## IMU Sensor Fusion With Simulink

Generate and fuse IMU sensor data using Simulink®. You can accurately model the behavior of an accelerometer, a gyroscope, and a

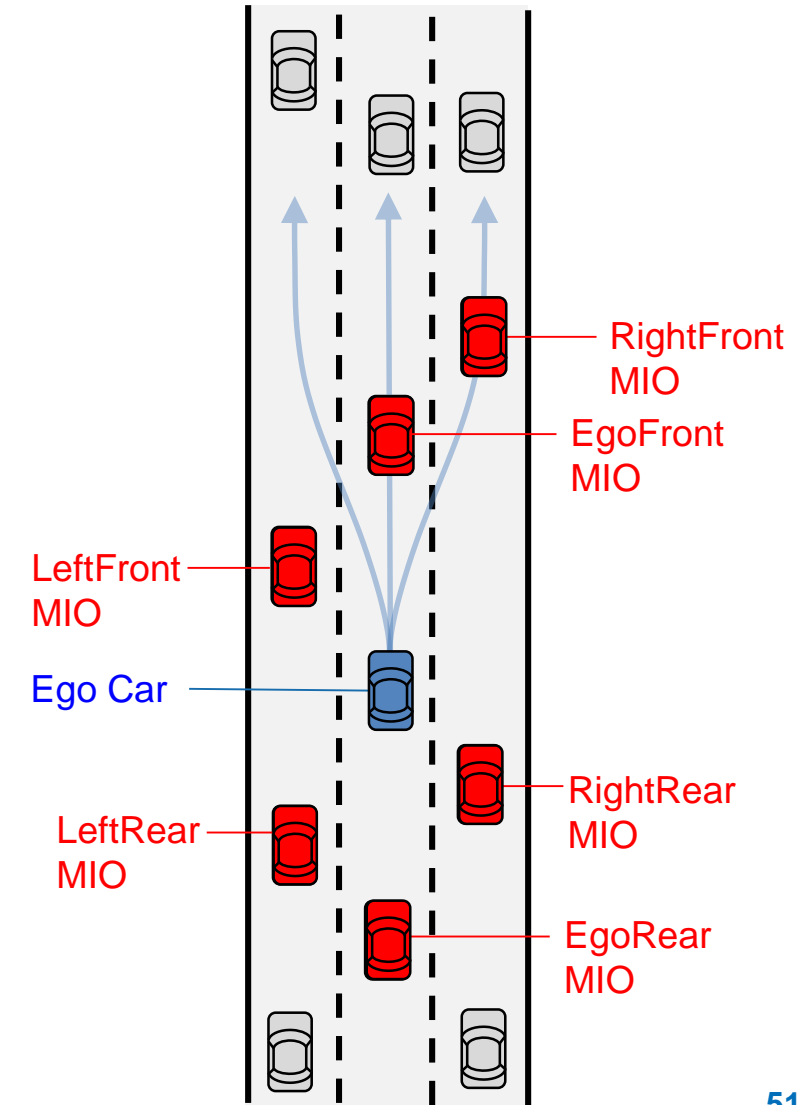
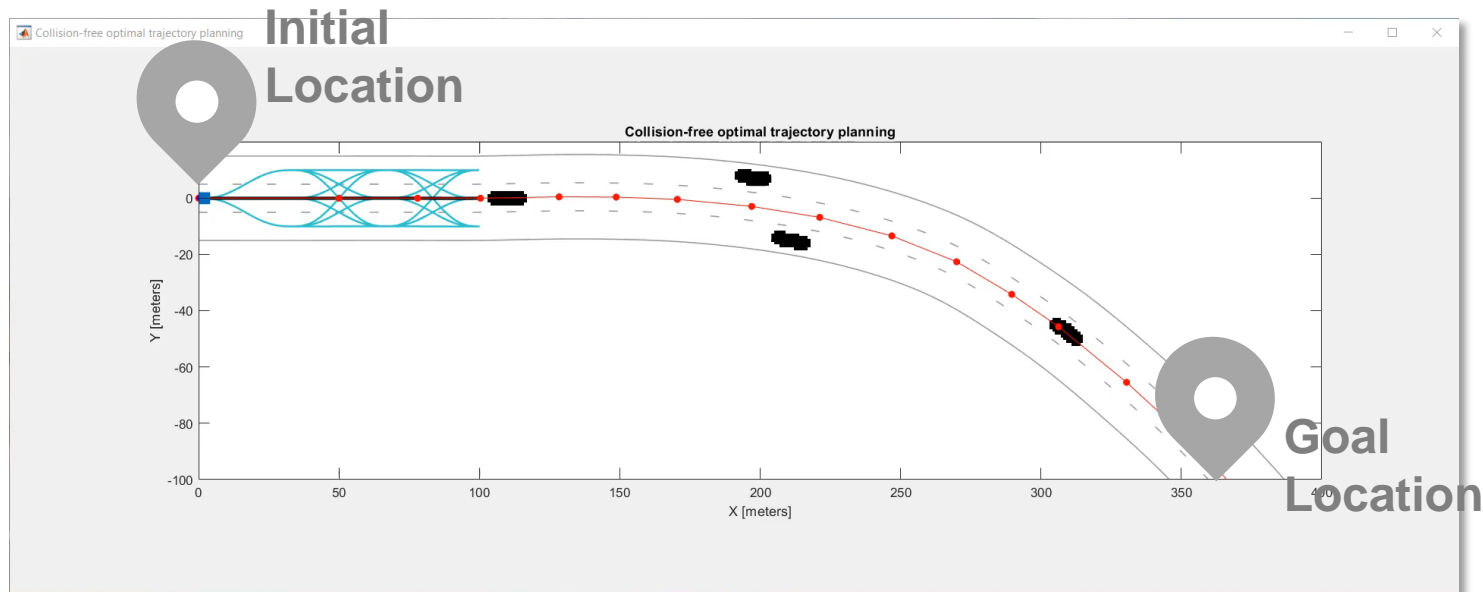
**R2020a** [Open Script](#)

# 目的地までのパスプラン・Decision Making

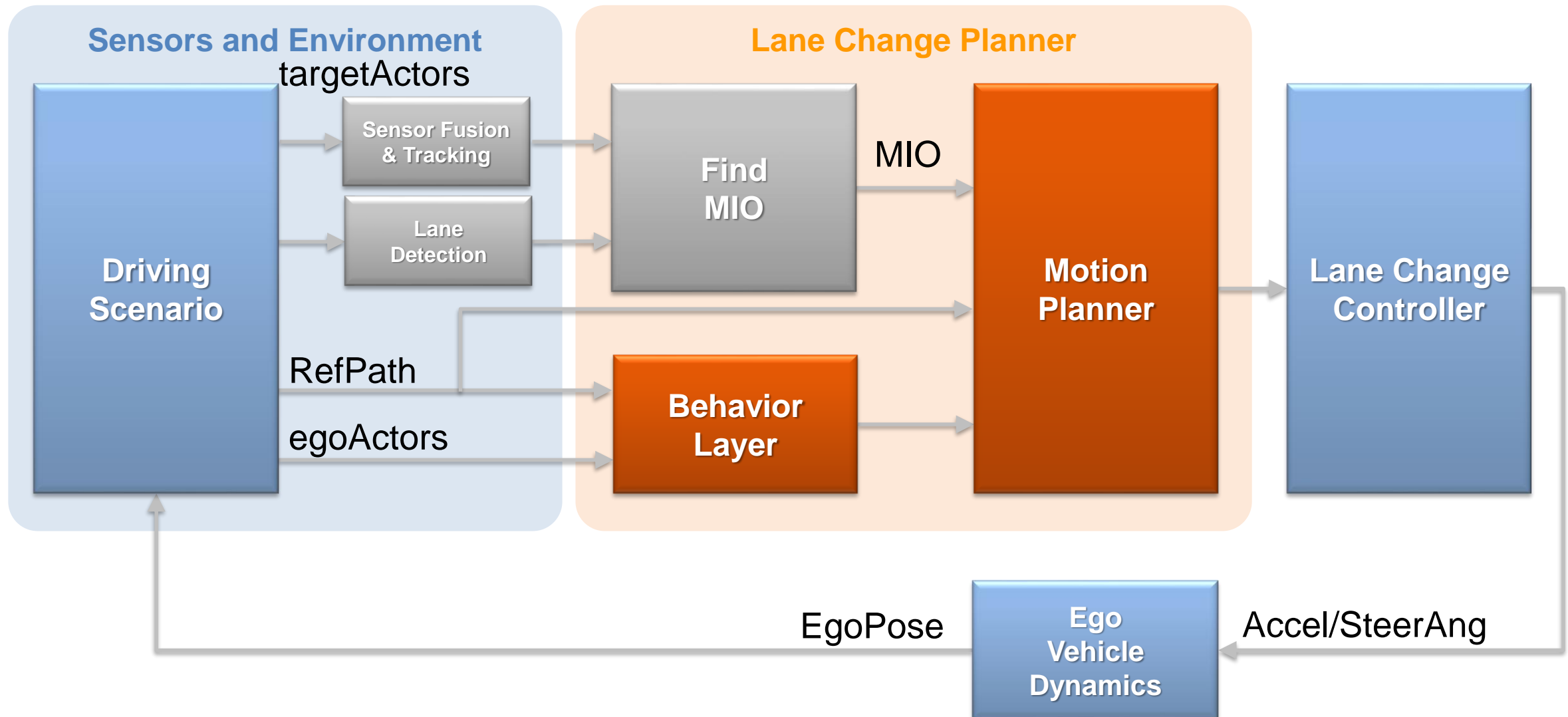


判断

Navigation Toolbox™

**Global  
Planning****Local Re-  
planning****Behavior  
Planning**

# 認知 + 判断 + 制御 統合モデルアーキテクチャ



# ローカルプランニング&制御アルゴリズムの開発例

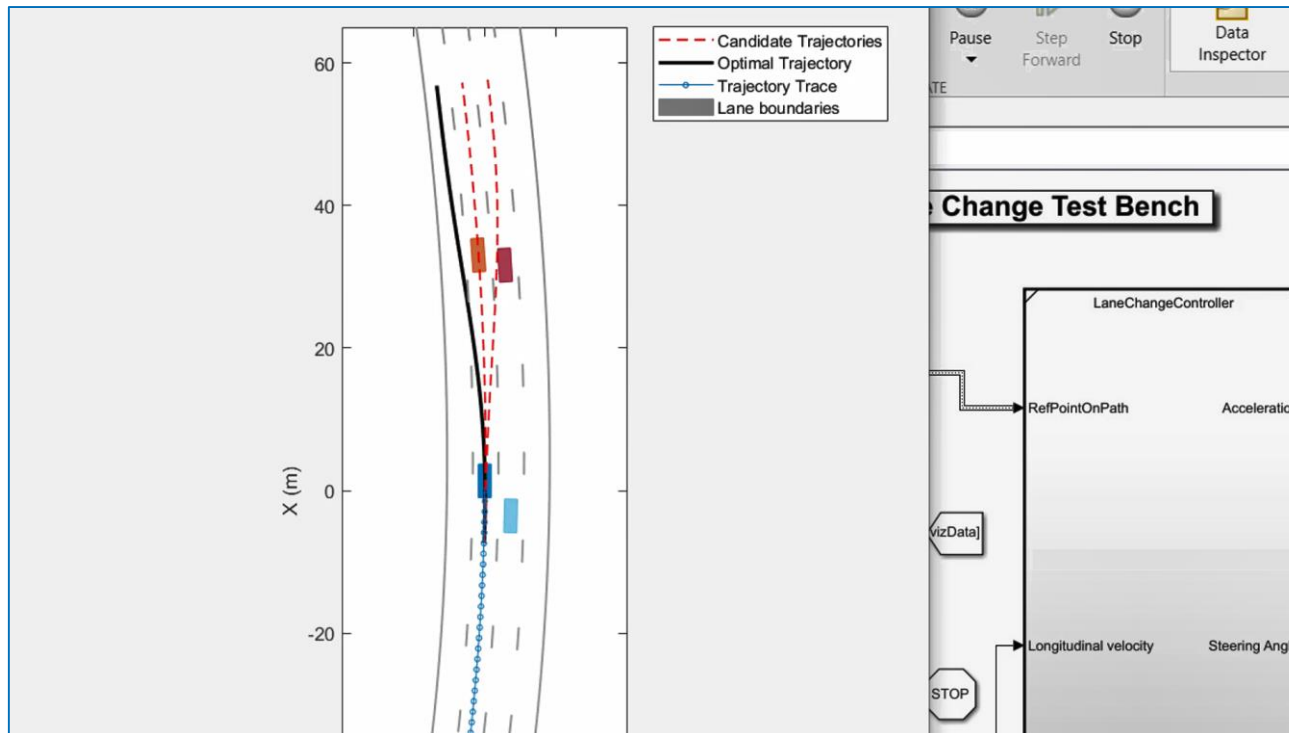
シナリオ  
定義

プランナー  
設計

制御アルゴ  
設計

ダイナミクス  
モデリング

結果の  
可視化



- Driving Scenarioで道路や車両、車両軌跡の設定
- 作成したシナリオをSimulink環境にインポート
- 既存制御系モデルとSimulink上で統合
- 設定した軌跡等シナリオ概要はDriving Scenario Designer上で可視化可能

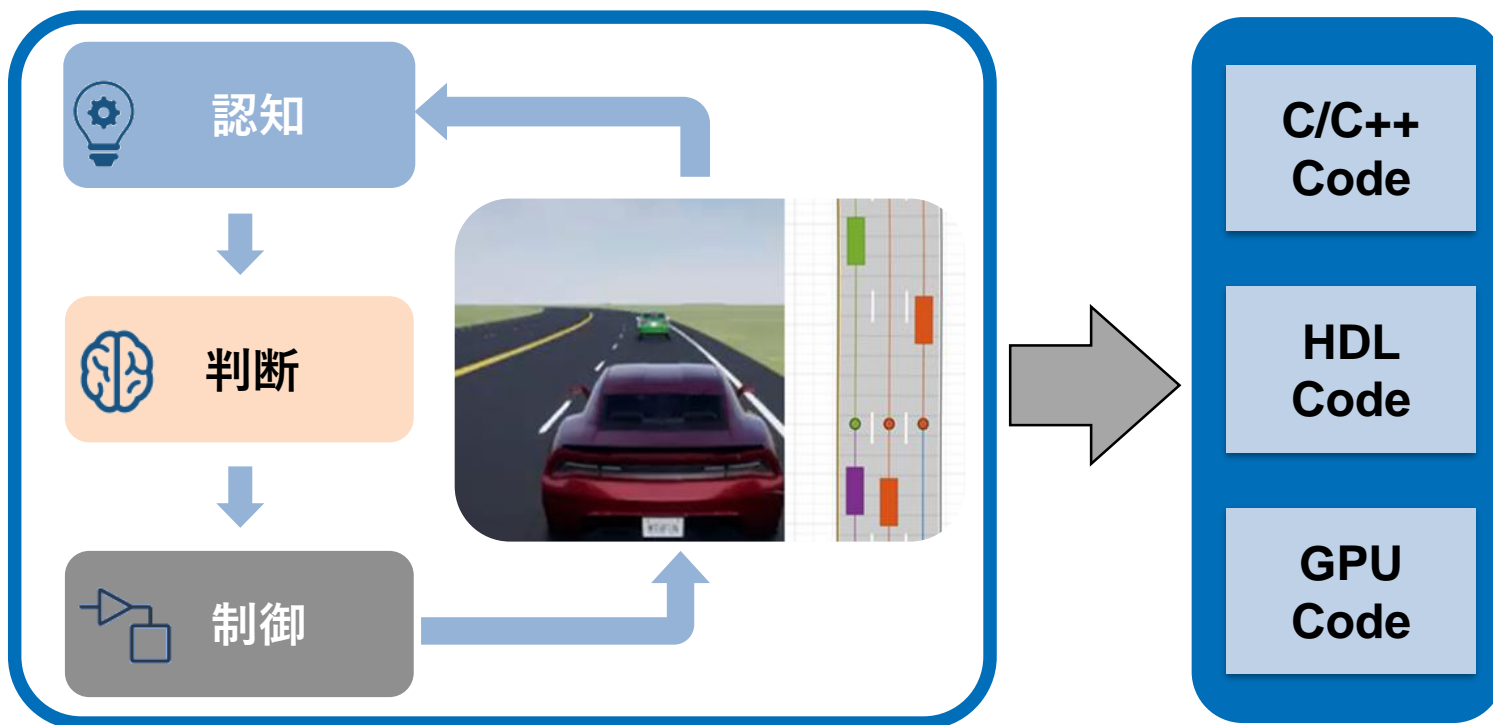
[Lane Change for Highway Driving](#)

Navigation Toolbox™

R2020a

# 組み込み機器への実装

Coders



Processors

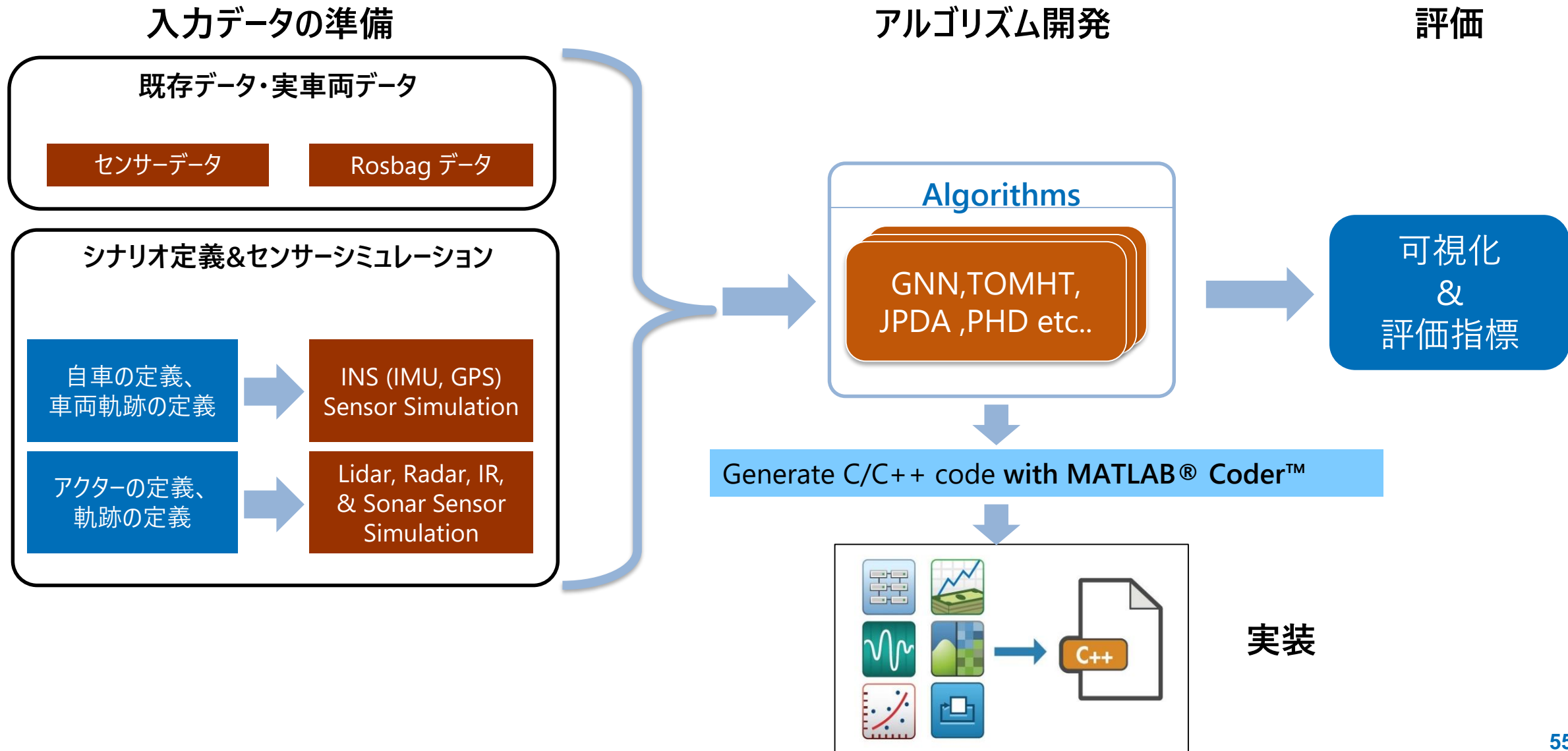


FPGAs

GPUs

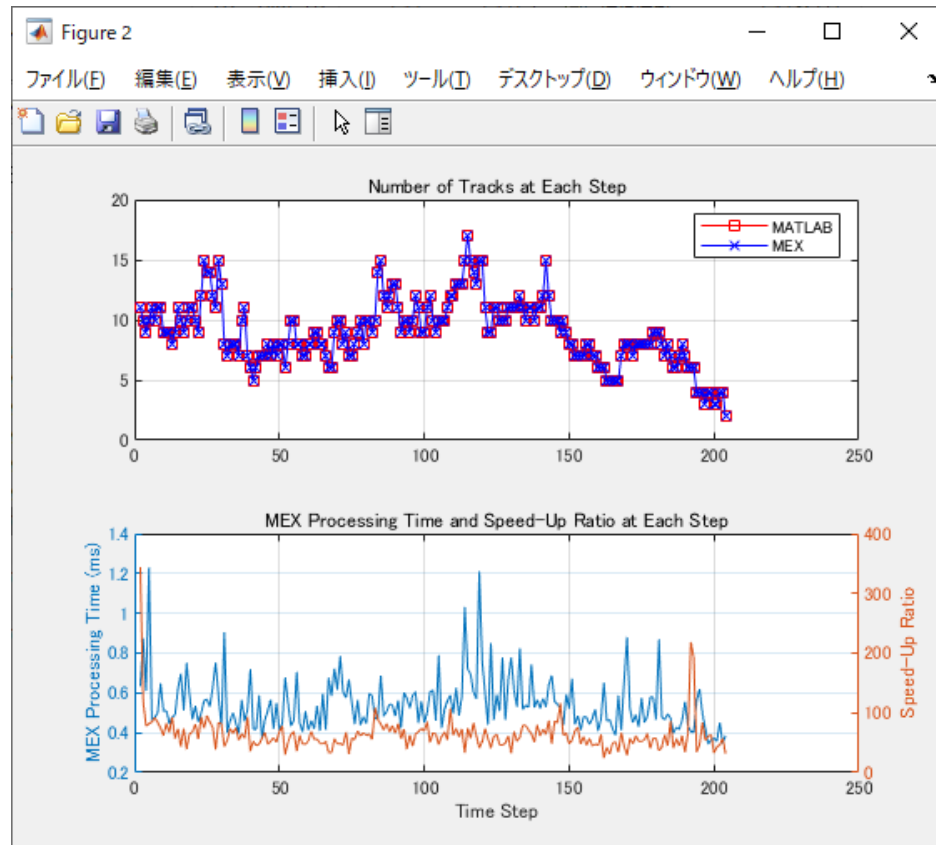


# センサーフュージョン&トラッキング アルゴリズム開発ワークフロー

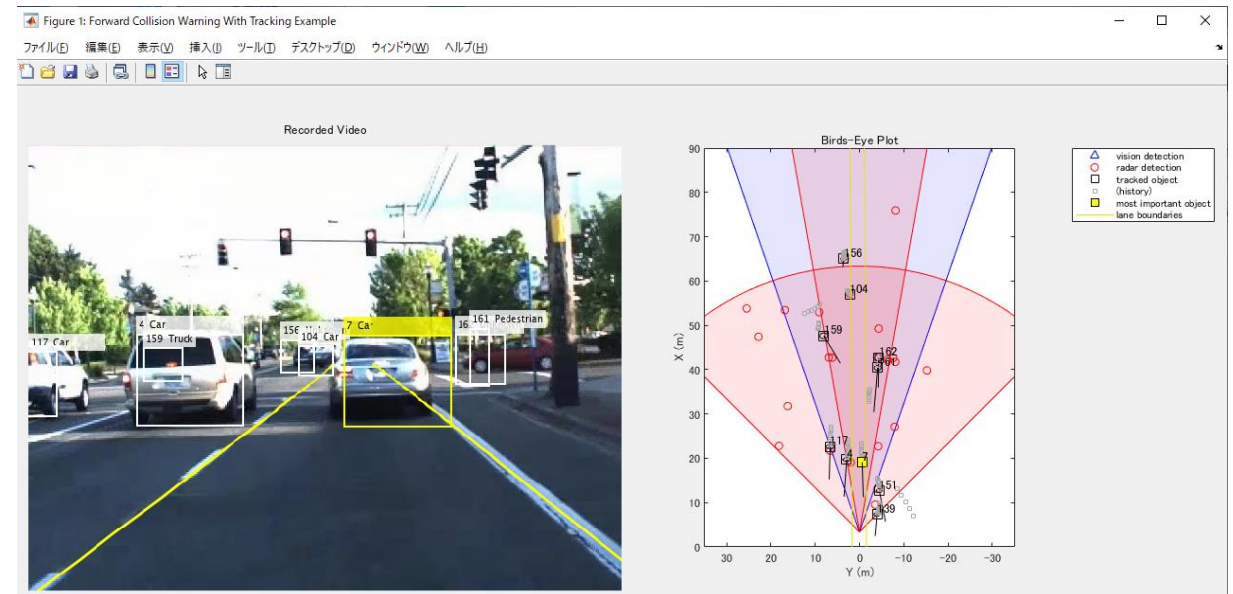


# multiObjectTrackerのコード生成

## MEXによる高速化



## Vision x 1, Radar x 2入力のSensor Fusion & Tracking



Generate C Code for a Tracker

Sensor Fusion and Tracking Toolbox™

MATLAB Coder®

## Key Takeaways :

### MATLAB/Simulinkによる自動運転アルゴリズム開発とシミュレーション ~センサーフュージョンを題材として~

- ✓ Driving Scenario Designerにより、様々なシナリオを定義
  - 様々なセンサモデル、オブジェクトリスト出力
- ✓ リファレンスとなるアルゴリズム群、評価用メトリクスの利用
- ✓ Simulinkブロックを利用し、既存の制御系モデルに統合
- ✓ コード生成に対応、高速化やラピッドプロト目的で利用可能

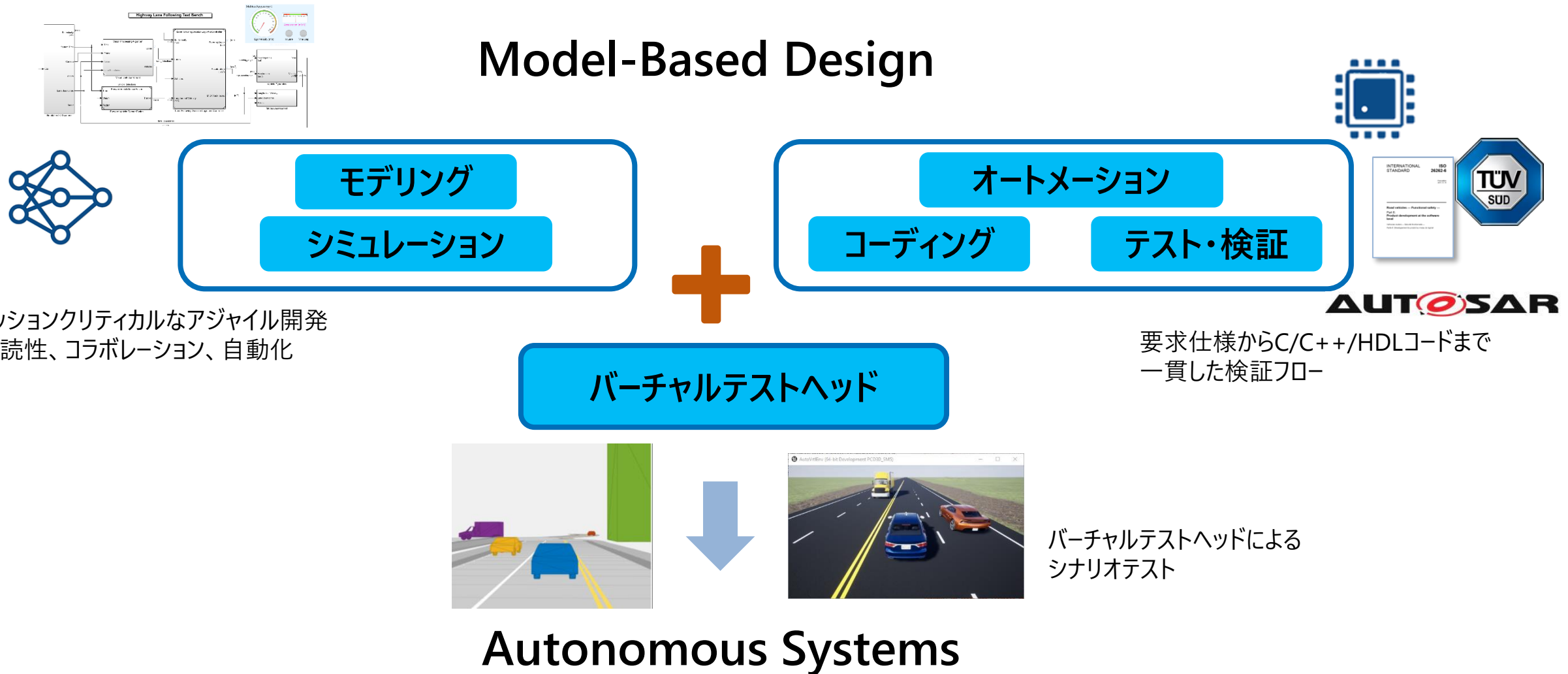


認知・判断・制御の各領域をSimulink上で統合、  
ADAS/AD領域においてもモデルベースデザインを実現!



# 自律システム・ADAS/自動運転への対応

## Model-Based Design for ADAS/AD



# プレゼンターへの質問

## 少々お待ちください

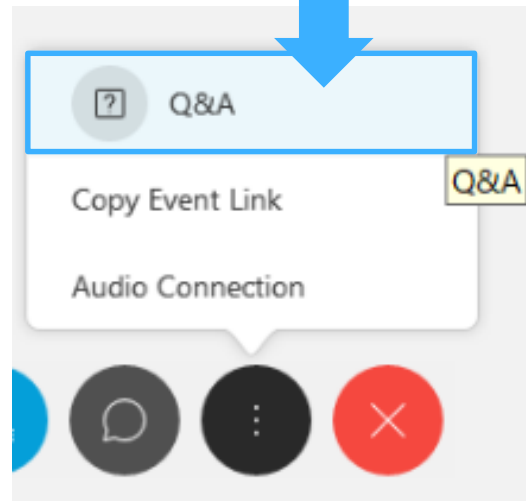
ご質問を確認する間、しばらく無音になりますが、ご遠慮なくQ&Aパネルよりご質問をお送りください。

Q&Aパネルを閉じてしまった場合は、画面下部中央のメニューバーから再度開いてください。

“...” ボタンをクリック



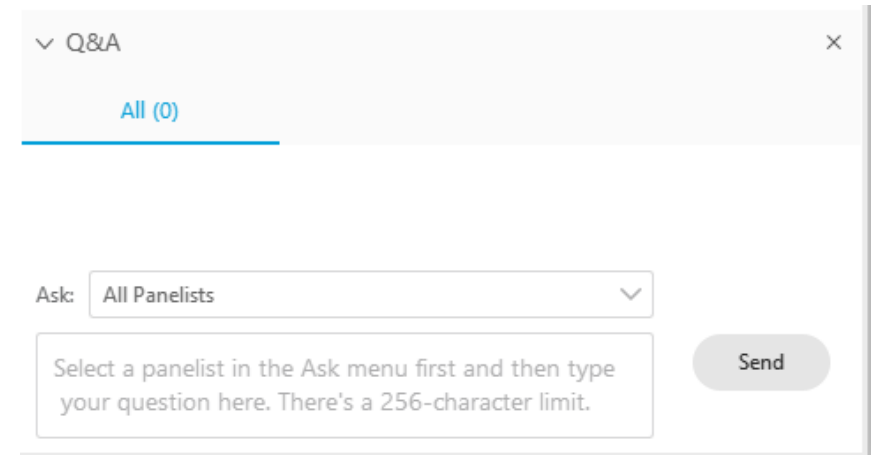
“Q&A”を選択



## 途中退出される場合

ログアウトするとブラウザーにアンケートが表示されますので、ご協力をお願いいたします。  
弊社からの回答をご希望の場合は、必ずお名前とご連絡先（Email）をご記入ください。

ボックスに質問を入力し“All Panelists (全てのパネリスト)”宛てに送信



本日のセミナーの録画版は、1週間ほどで [jp.mathworks.com/videos](https://jp.mathworks.com/videos) に掲載されます。  
配布可能な資料がある場合は、該当ページの“コードとリソース”タブからダウンロードいただけます。



© 2020 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [www.mathworks.com/trademarks](http://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

