OPENSCENARIO USE CASE WORKSHOP

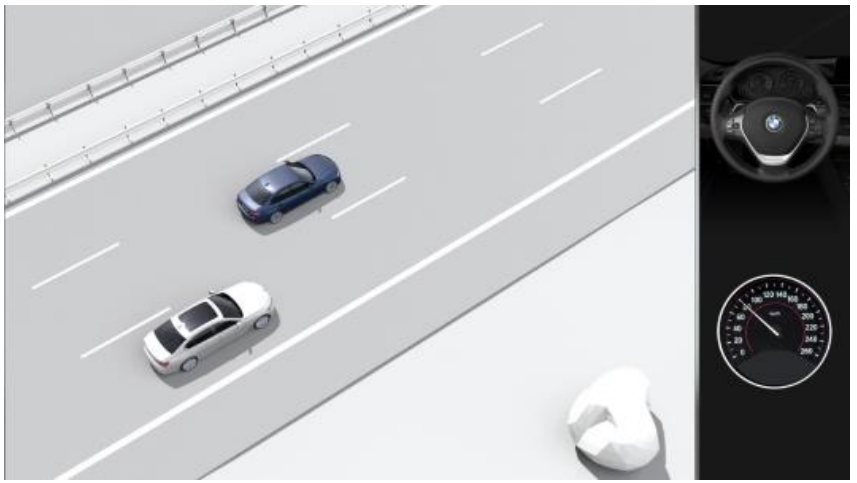
USE CASES & REQUIREMENTS @ BMW

Ludwig Friedmann | September 17th, 2018

BMW GROUP

THE NEXT 100 YEARS

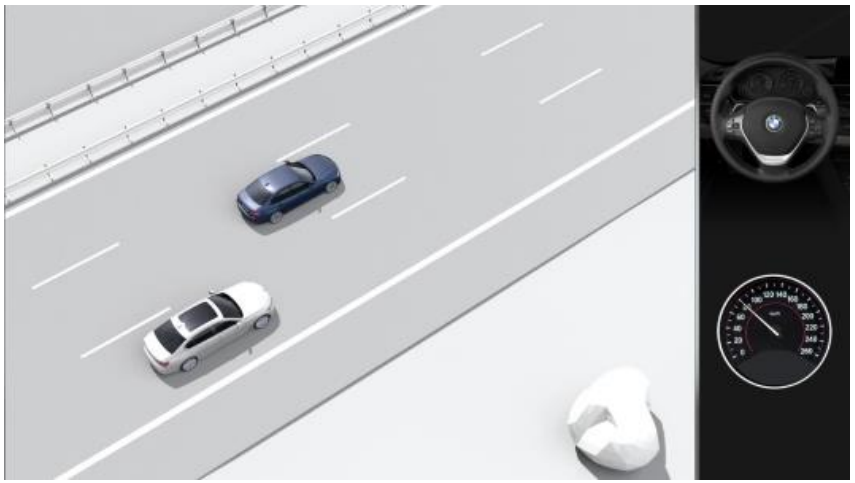
USE CASES



USE CASES

1. Effectiveness Analysis

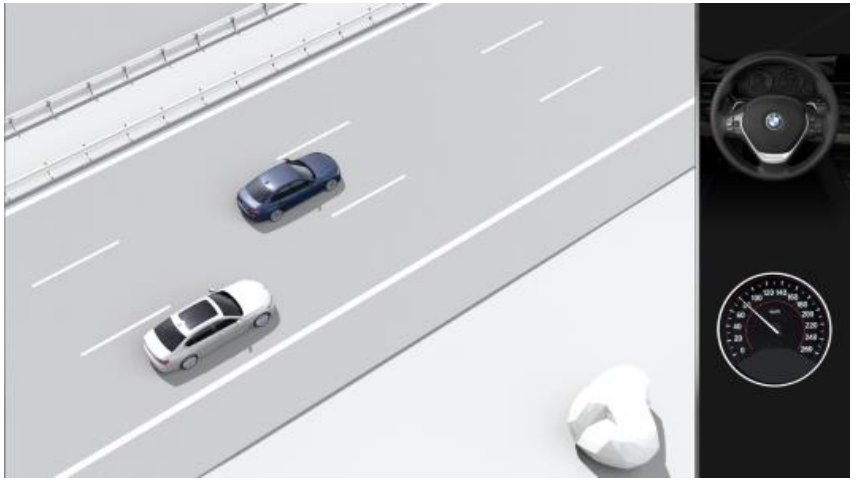
- Minor usage of scripting, traffic represented by statistical models
- Limited scenario size and complexity
- Manual scenario creation in text editor



USE CASES

1. **Effectiveness Analysis**
   - Minor usage of scripting, traffic represented by statistical models
   - Limited scenario size and complexity
   - Manual scenario creation in text editor

2. **Driving Simulation**
   - Human driver, scripted scenarios and complex conditions
   - Potentially large scenarios
   - Manual scenario creation using graphical editors and scripting (text editor)

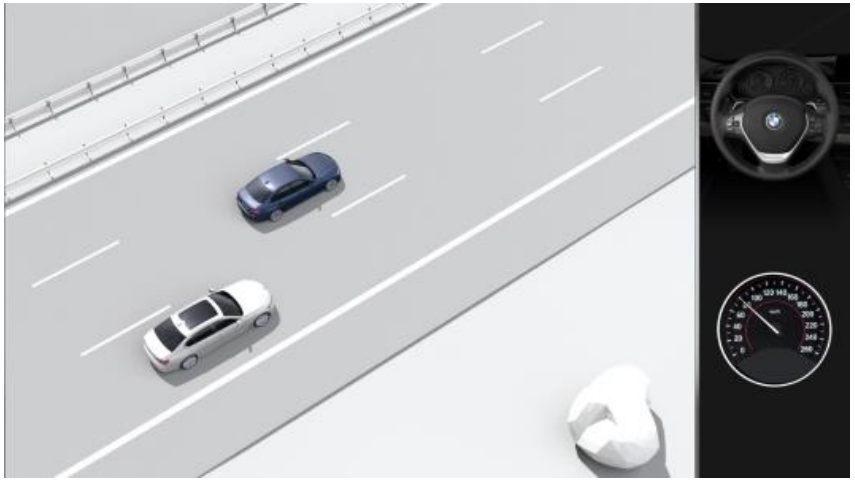


USE CASES

1. **Effectiveness Analysis**
   - Minor usage of scripting, traffic represented by statistical models
   - Limited scenario size and complexity
   - Manual scenario creation in text editor

2. **Driving Simulation**
   - Human driver, scripted scenarios and complex conditions
   - Potentially large scenarios
   - Manual scenario creation using graphical editors and scripting (text editor)

3. **(Cluster-)Simulation / Test of AD Function**
   - Traffic modelled by fixed trajectories
   - Varying scenario size
   - Scenario creation by upstream software



USE CASES

1. **Effectiveness Analysis**
   - Minor usage of scripting, traffic represented by statistical models
   - Limited scenario size and complexity
   - Manual scenario creation in text editor

2. **Driving Simulation**
   - Human driver, scripted scenarios and complex conditions
   - Potentially large scenarios
   - Manual scenario creation using graphical editors and scripting (text editor)

3. **(Cluster-)Simulation / Test of AD Function**
   - Traffic modelled by fixed trajectories
   - Varying scenario size
   - Scenario creation by upstream software

4. **Development-Related Testing**
   - Limited scenario size
   - Varying scenario complexity
   - Scenario creation by graphical / text editor

CONSOLIDATED FEATURE SET

1. Simulation Coordinate Systems
2. Geodetic Coordinate Systems
3. Map Reference
4. Trajectories and Routes
5. Traffic Participants
6. Objects
7. Traffic Signs and Signals
8. Environment
9. Conditions and Functions
10. Event Definitions
11. Event Handling and Actions
12. Stochastics Module

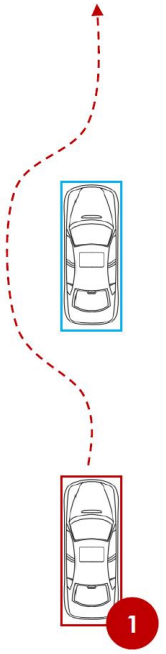
11. Generative Scenarios
12. Validity Evaluation
13. Library Concept
14. Error Handling

SIMPLE EXAMPLE: OVERTAKER

200 lines of XML code in OpenSCENARIO

INTUITIVE HIGH-LEVEL DESCRIPTION LANGUAGE

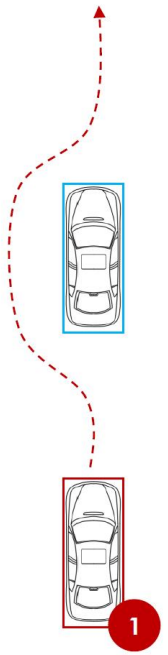
```
Scenario SimpleOvertaker:

  Act Overtaking:
    @init:
      CreateVehicle(Ego,EgoProto,xyz=0m|0m|0m,speed=80km/h)
      CreateVehicle(Overtaker,OvertakerProto,xyz=0m|Ego.y-100m|0m,speed=100km/h)
    @Distance(Ego,Overtaker) <= 30m % OnceOnly:
      LaneChange(Overtaker,lane=left,model=Sinusoidal(time=5s))
    @DistanceAfterPassing(Ego,Overtaker) >= 5m % OnceOnly:
      LaneChange(Overtaker,lane=right,model=Sinusoidal(time=5s))

  Prototypes:
    EgoProto: Vehicle(geometry="mycar.obj",color="blue")
    OvertakerProto: Vehicle(geometry="traffic.obj",color="red")

  Resources:
    Map: OpenDrive("OpenDriveMap.xodr")
```



INTUITIVE HIGH-LEVEL DESCRIPTION LANGUAGE

```
Scenario SimpleOvertaker:

  Act Overtaking:
    @init:
      CreateVehicle(Ego,EgoProto,xyz=0m|0m|0m,speed=80km/h)
      CreateVehicle(Overtaker,OvertakerProto,xyz=0m|Ego.y-100m|0m,speed=100km/h)
    @Distance(Ego,Overtaker) <= 30m % OnceOnly:
      LaneChange(Overtaker,lane=left,model=Sinusoidal(time=5s))
    @DistanceAfterPassing(Ego,Overtaker) >= 5m % OnceOnly:
      LaneChange(Overtaker,lane=right,model=Sinusoidal(time=5s))

  Prototypes:
    EgoProto: Vehicle(geometry="mycar.obj",color="blue")
    OvertakerProto: Vehicle(geometry="traffic.obj",color="red")

  Resources:
    Map: OpenDrive("OpenDriveMap.xodr")
```

Acts

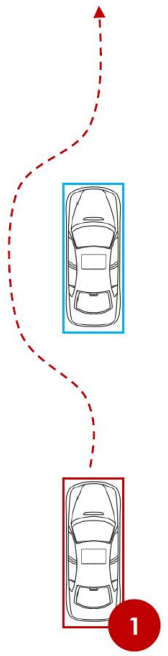


INTUITIVE HIGH-LEVEL DESCRIPTION LANGUAGE

```
Scenario SimpleOvertaker:

  Act Overtaking:
    @init:
      CreateVehicle(Ego,EgoProto,xyz=0m|0m|0m,speed=80km/h)
      CreateVehicle(Overtaker,OvertakerProto,xyz=0m|Ego.y-100m|0m,speed=100km/h)
    @Distance(Ego,Overtaker) <= 30m % OnceOnly:
      LaneChange(Overtaker,lane=left,model=Sinusoidal(time=5s))
    @DistanceAfterPassing(Ego,Overtaker) >= 5m % OnceOnly:
      LaneChange(Overtaker,lane=right,model=Sinusoidal(time=5s))

  Prototypes:
    EgoProto: Vehicle(geometry="mycar.obj",color="blue")
    OvertakerProto: Vehicle(geometry="traffic.obj",color="red")

  Resources:
    Map: OpenDrive("OpenDriveMap.xodr")
```

Acts

Triggers



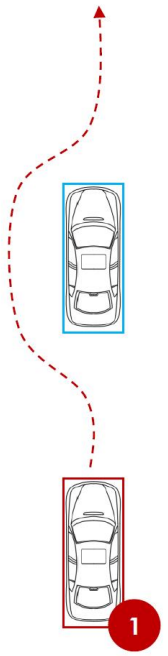
INTUITIVE HIGH-LEVEL DESCRIPTION LANGUAGE

```
Scenario SimpleOvertaker:

  Act Overtaking:
    @init:
      CreateVehicle(Ego,EgoProto,xyz=0m|0m|0m,speed=80km/h)
      CreateVehicle(Overtaker,OvertakerProto,xyz=0m|Ego.y-100m|0m,speed=100km/h)
    @Distance(Ego,Overtaker) <= 30m % OnceOnly:
      LaneChange(Overtaker,lane=left,model=Sinusoidal(time=5s))
    @DistanceAfterPassing(Ego,Overtaker) >= 5m % OnceOnly:
      LaneChange(Overtaker,lane=right,model=Sinusoidal(time=5s))

  Prototypes:
    EgoProto: Vehicle(geometry="mycar.obj",color="blue")
    OvertakerProto: Vehicle(geometry="traffic.obj",color="red")

  Resources:
    Map: OpenDrive("OpenDriveMap.xodr")
```

Acts

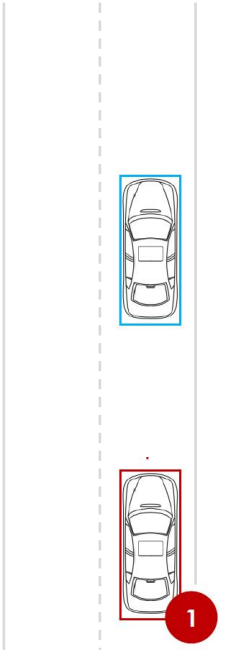
Triggers

Actions



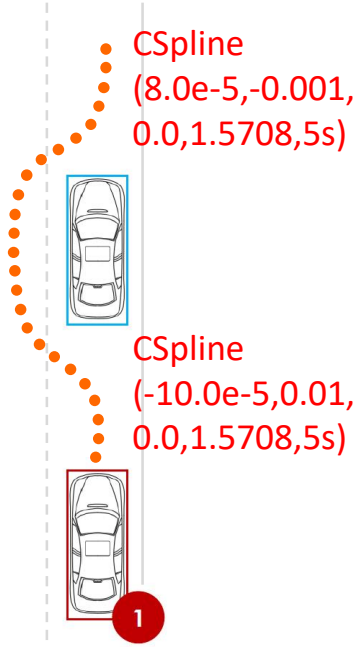
DERIVATION OF A LOW-LEVEL DESCRIPTION

```
Scenario SimpleOvertaker:

  Act Overtaking:
    @init:
      CreateVehicle(Ego,EgoProto,xyz=0m|0m|0m,speed=80km/h)
      CreateVehicle(Overtaker,OvertakerProto,xyz=0m|Ego.y-100m|0m,speed=100km/h)
    @Distance(Ego,Overtaker) <= 30m % OnceOnly:
      LaneChange(Overtaker,lane=left,model=Sinusoidal(time=5s))
    @DistanceAfterPassing(Ego,Overtaker) >= 5m % OnceOnly:
      LaneChange(Overtaker,lane=right,model=Sinusoidal(time=5s))

  Prototypes:
    EgoProto: Vehicle(geometry="mycar.obj",color="blue")
    OvertakerProto: Vehicle(geometry="traffic.obj",color="red")

  Resources:
    Map: OpenDrive("OpenDriveMap.xodr")
```



DERIVATION OF A LOW-LEVEL DESCRIPTION

```
Scenario SimpleOvertaker:

  Act Overtaking:
    @init:
      CreateVehicle(Ego,EgoProto,xyz=0m|0m|0m,speed=80km/h)
      CreateVehicle(Overtaker,OvertakerProto,xyz=0m|Ego.y-100m|0m,speed=100km/h)
    @Distance(Ego,Overtaker) <= 30m % OnceOnly:
      LaneChange(Overtaker,lane=left,model=Sinusoidal(time=5s))
    @DistanceAfterPassing(Ego,Overtaker) >= 5m % OnceOnly:
      LaneChange(Overtaker,lane=right,model=Sinusoidal(time=5s))

  Prototypes:
    EgoProto: Vehicle(geometry="mycar.obj",color="blue")
    OvertakerProto: Vehicle(geometry="traffic.obj",color="red")

  Resources:
    Map: OpenDrive("OpenDriveMap.xodr")
```

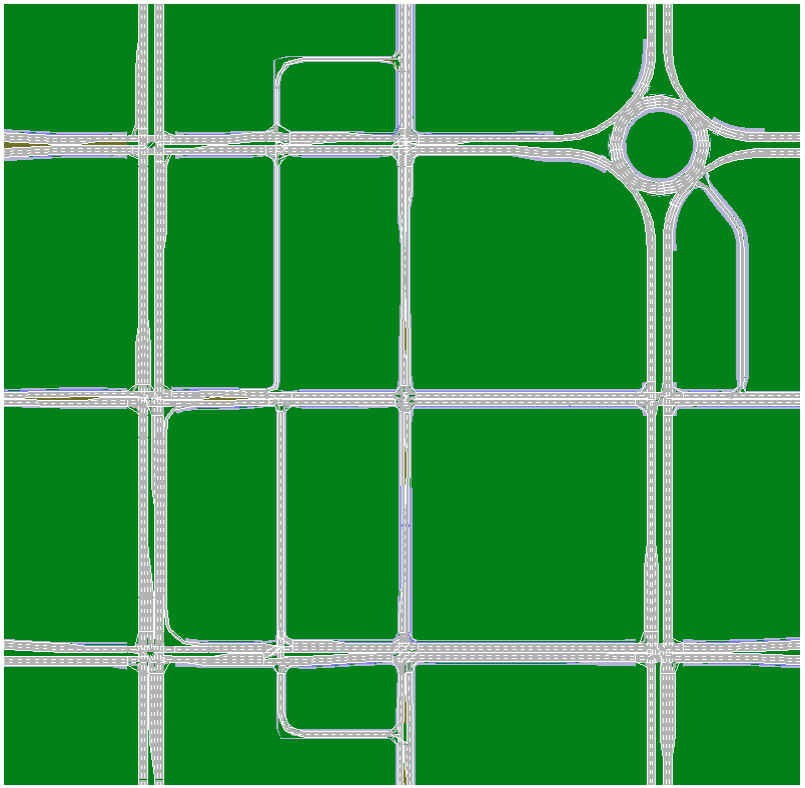


Reference implementation required for transformation into machine readable, mathematical description

PATTERN BASED SCENARIO GENERATION

- Manual scenario creation can not scale to the requirements posed by AD function development & validation

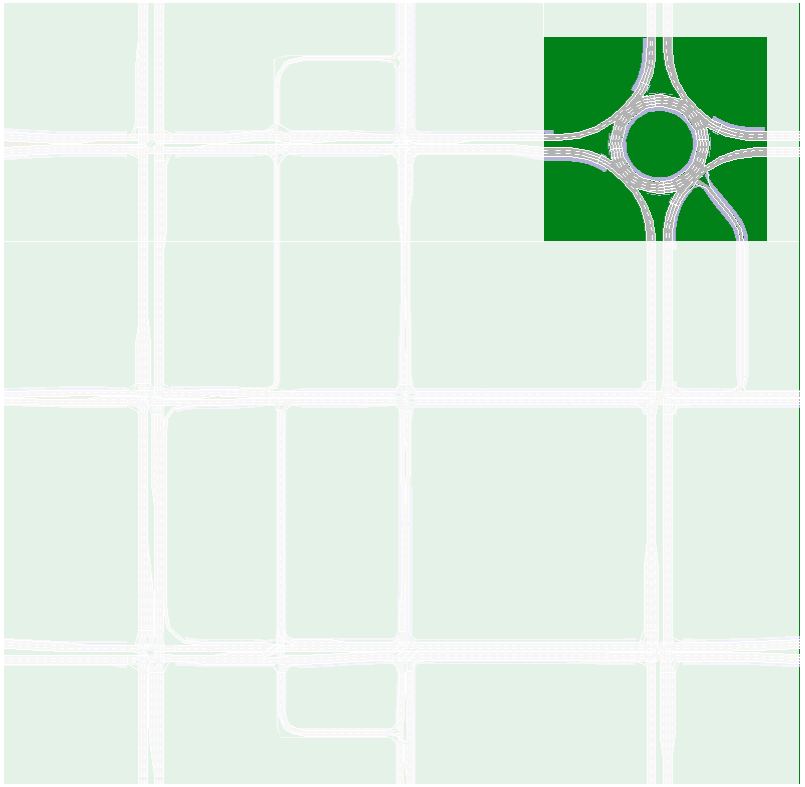
- Definition and calculation of trajectories can be supported by automated algorithms



PATTERN BASED SCENARIO GENERATION

- Manual scenario creation can not scale to the requirements posed by AD function development & validation

- Definition and calculation of trajectories can be supported by automated algorithms



Search algorithms based on pattern matching may open up new paths to describe large scale scenarios.

CONSOLIDATED REQUIREMENTS

- Scenarios of varying size and complexity have to be editable in text editors. This requires an abstract, **intuitive high-level description language**.

- The interpretation of abstract content requires models. **Reference implementations** enable the transformation of this content into mathematically exact descriptions.

- This mathematical description shall be deterministically reproducible using a machine-readable **low-level description language**.

- Map based **pattern matching** may open up new ways in scenario description.