HIGH-LEVEL EXTENSION FOR OPENSCENARIO

Ludwig Friedmann | November 13th, 2018

BMW GROUP THE NEXT 100 YEARS

SIMPLE EXAMPLE: OVERTAKER

200 lines of XML code in OpenSCENARIO

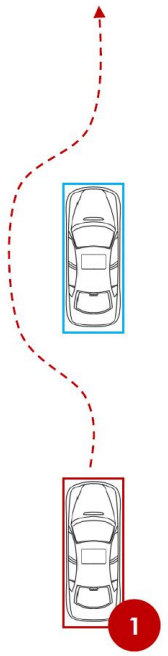
INTUITIVE HIGH-LEVEL DESCRIPTION LANGUAGE

```
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")
```

Acts

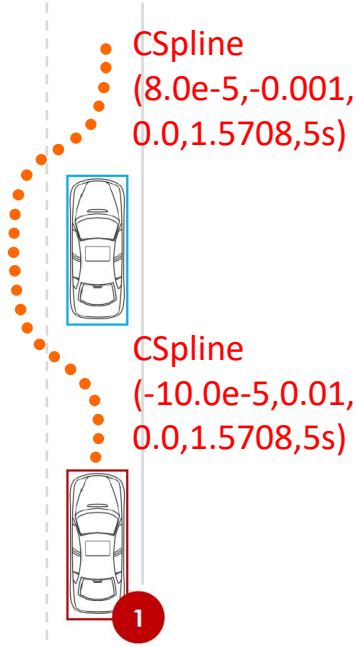
Triggers

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")
```



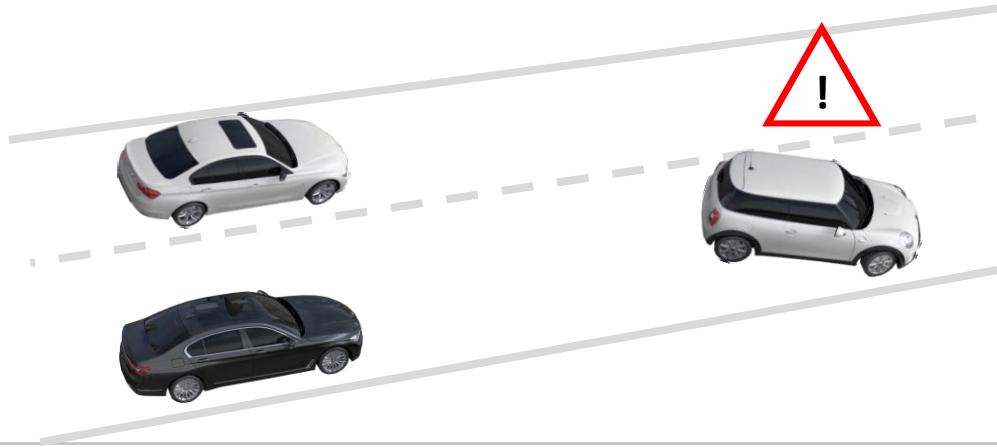
Reference implementation required for transformation into machine readable, mathematical description

ADVANCED EXAMPLE: CRITICAL TAKE OVER REQUEST

- Accident on rightmost lane
- Driving function has to initiate critical take over request
- Left lane is blocked by vehicle with slightly higher velocity

Scenario

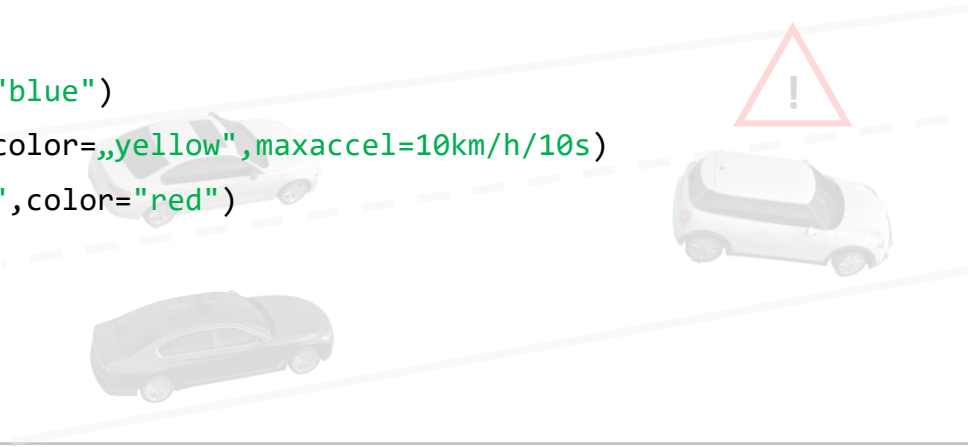
- Accident 2000 m ahead of Ego startposition
- Blocker initially located 1000 m behind Ego on rightmost lane
- With Ego 1000 m ahead of Accident, Blocker
  - changes to left lane
  - adapts velocity to catch up with Ego just in front of the accident
- When almost reaching Ego, Blocker accelerates by 10 km/h



ADVANCED EXAMPLE: CRITICAL TAKE OVER REQUEST

```
Scenario BlockedTOR:
  Act AccidentTOR:
    @init:
      CreateVehicle(Ego,EgoProto,xyz=0m|0m|0m,speed=130km/h)
      CreateVehicle(Blocker,BlockerProto,xyz=0m|Ego.y-1000m|0m,speed=180km/h,model=AdvancedDriver(…))
      CreateVehicle(Accident,AccidentProto,xyz=0m|Ego.y+2000m|0m,speed=0km/h)
    @Distance(Ego,Accident) <= 1000m % OnceOnly:
      LaneChange(Blocker,lane=+1,model=Sinusoidal(time=5s))
      SpeedChange(Blocker,SpeedToReach(Blocker,Ego,Accident.y-100m))
    @Distance(Blocker,Ego) <= 30m % OnceOnly:
      SpeedChange(Blocker,Ego.velocity.y+10km/h)
  Prototypes:
    EgoProto: Vehicle(geometry="mycar.obj",color="blue")
    BlockerProto: Vehicle(geometry=„blocker.obj",color=„yellow",maxaccel=10km/h/10s)
    AccidentProto: Vehicle(geometry=„accident.obj",color="red")
  Resources:
    Map: OpenDrive("OpenDriveMap.xodr")
```

Human drivers require smarter scenarios :)

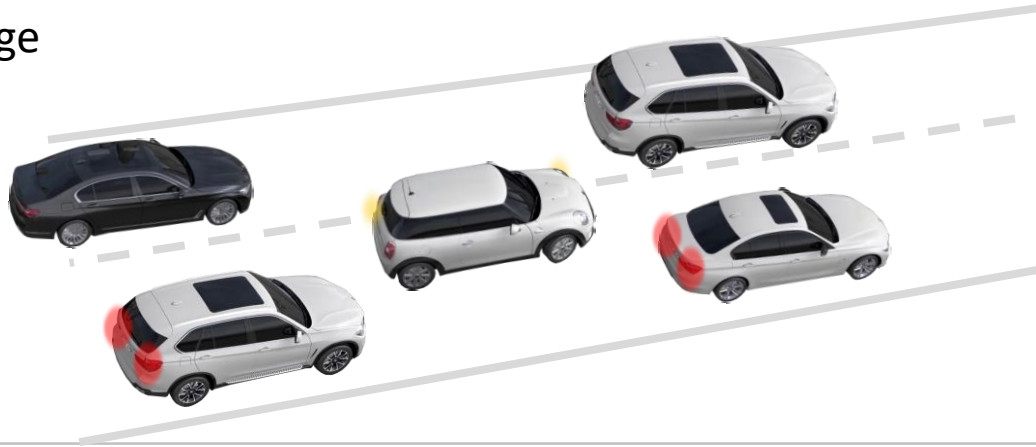


ADVANCED EXAMPLE: QUEUE SWITCHERS

– Ego drives at moderate speed on left lane, keeping free space to vehicle in front of it
– From the rightmost lane, several (slow) vehicles try to enter this gap

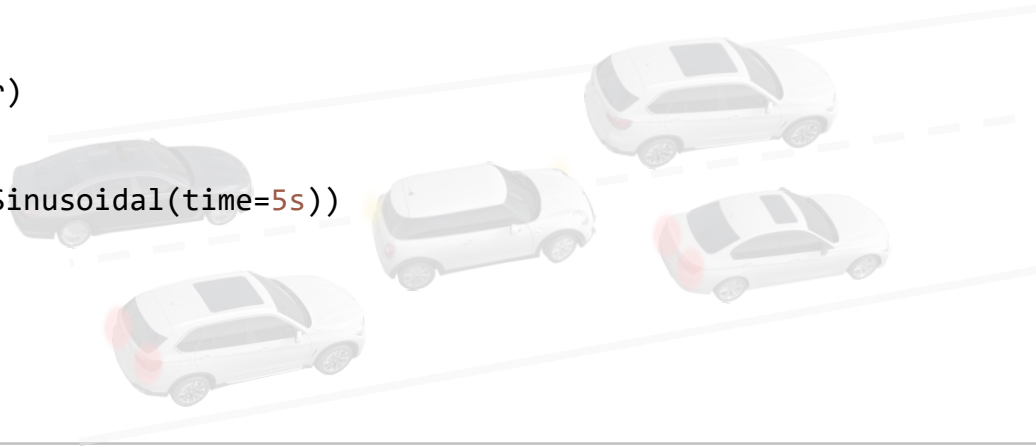
Scenario

– Ego and FrontBlocker drive at const. speed (30 km/h), distance 100 m
– 30 QueueSwitchers (QS) travel 15 m apart of each other at 10 km/h on rightmost lane
– If the gap is large enough, the closest QS in front of Ego is selected to
  – laterally move by 0.3 m
  – use indicators to announce lane change
  – perform lane change



ADVANCED EXAMPLE: QUEUE SWITCHERS

```
Scenario QueueSwitcher:
  Act QueueSwitcher($rightcount=30,$rightspeed=10km/h,$latpos=0.3,$latpostime=1s,$blinktime=2s):
    @init:
      CreateVehicle(Ego,EgoProto,xyz=0m|0m|0m,speed=30km/h)
      CreateVehicle(Front,FrontProto,xyz=0m|Ego.y+100m|0m,speed=30km/h)
      repeat $n in [1:$rightcount] by 1:
        CreateVehicle(RightQueue$n,RightQueueProto,xyz=5m|Ego.y+$n*15m|0m,speed=$rightspeed)
    @Headway(Ego,Front) >= 2s % OnceOnly:
      select $next_queue in RightQueue* with DistanceBeforePassing($next_queue,Ego)>15m
             minimize Distance($next_queue,Ego):
        LateralChange($next_queue,x=-$latpos,t=$latpostime)
        Delay(2s)
        SetLight($next_queue,LeftTurnIndicator)
        Delay($blinktime)
        LaneChange($next_queue,lane=+1,model=Sinusoidal(time=5s))
```



FINAL THOUGHTS

- In our daily work, users approach us with a set of powerpoint slides to describe critical situations.
- Based on these slides, we generate scenario descriptions for driving simulators or cluster simulation.
- We show the scenario description to the requesters and ask for a review.
- Unfortunately, today, the description language is too complex, he can neither check the correctness by means of the scenario description nor in simulation.
- **With an high-level extension of OpenSCENARIO, the user can understand the scenario, correct it and write it himself in the next case.**