Dipl.-Ing. Konstantin Blenz, Dipl.-Ing. Christian Siebke
Fakultät Verkehrswissenschaften „Friedrich List“
Institut für Automobiltechnik – IAD
Lehrstuhl für Kraftfahrzeugtechnik – LKT

OpenPASS Architecture Commitee Meeting

Modular driver-architecture & demands on the framework

Munich, 04.07.2019

Agenda

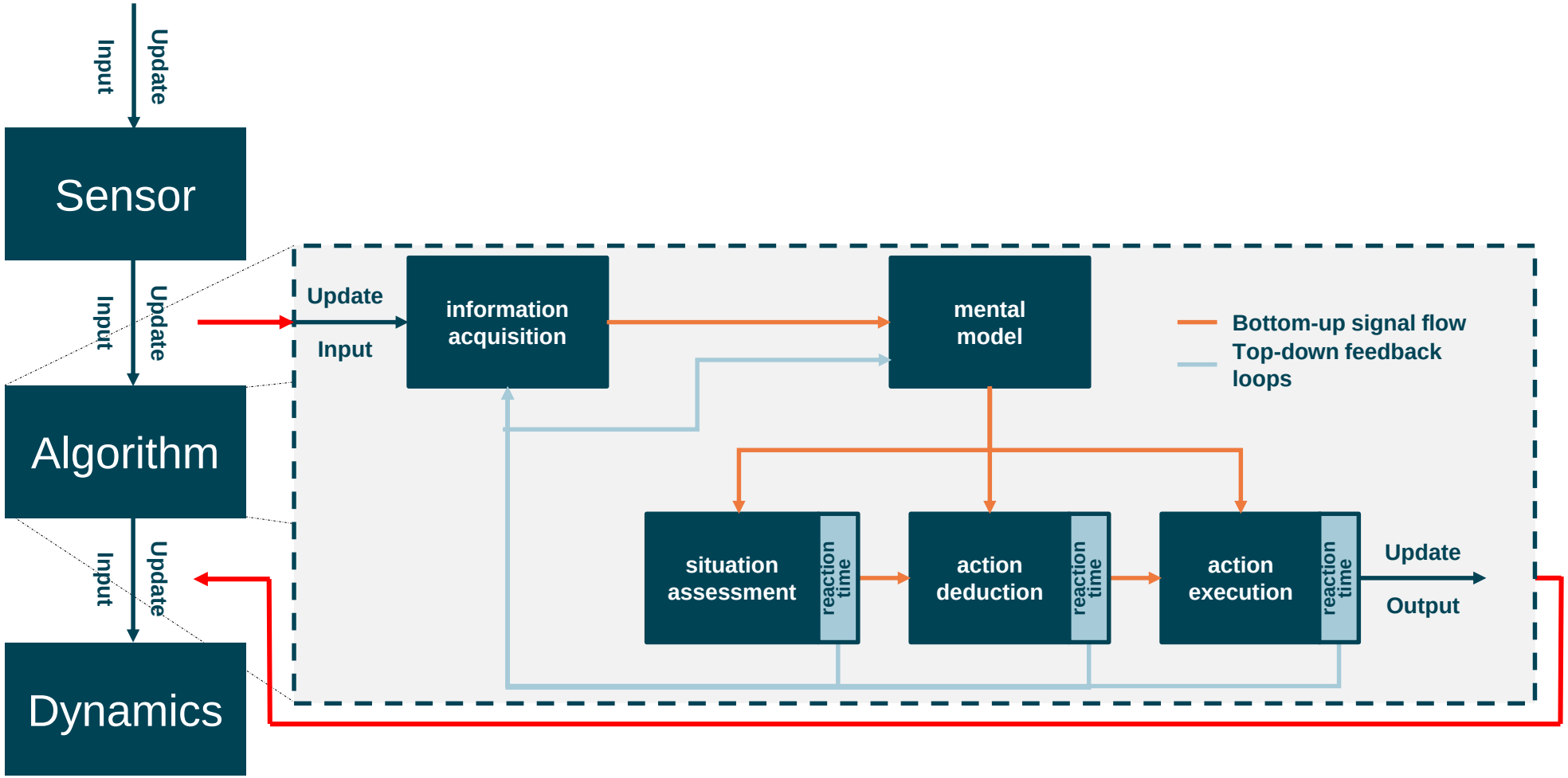
1. Modular driver-architecture
2. Usecase and dependencies on the framework

Agenda

1. Modular driver-architecture
2. Usecase and dependencies on the framework

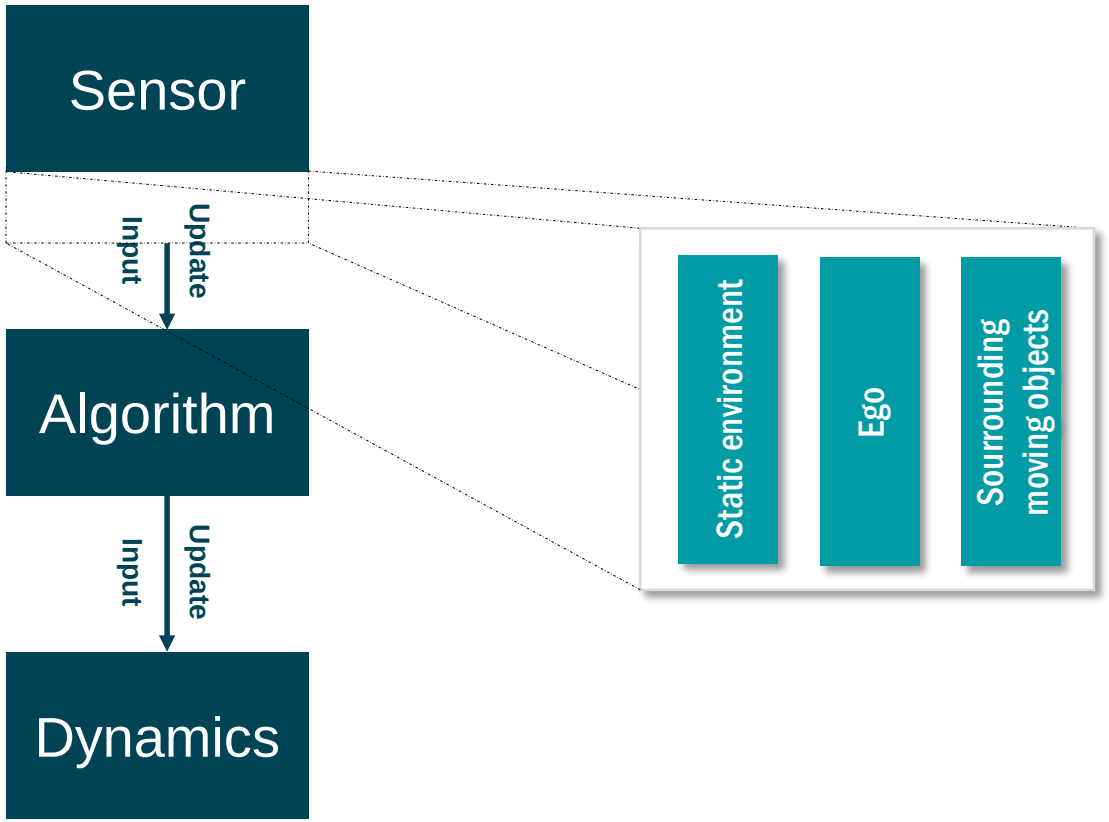
Modular driver-architecture

Overview



Modular driver-architecture

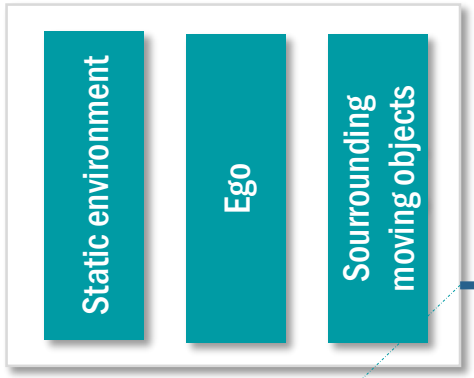
Update Input



Container structure
containing object information
e.g. size, dynamics

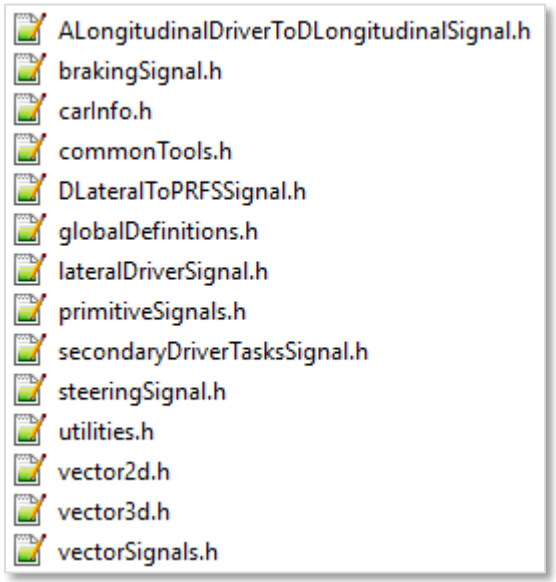
Modular driver-architecture

Signal classes



information acquisition

New signal class for variable container structures necessary



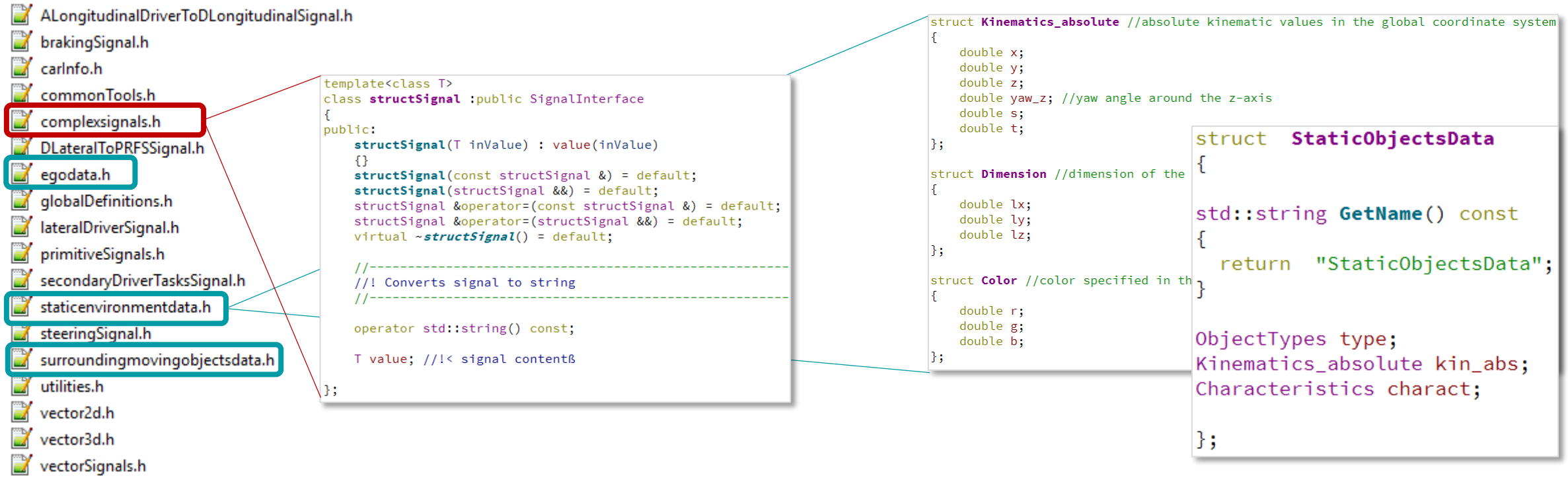
Example

```
const std::shared_ptr<LateralDriverSignal const> signal = std::dynamic_pointer_cast<LateralDriverSignal const>(data);

in_currCurvature = signal->currCurvature;
in_vLatEgo = signal->vLatEgo;
in_vLongEgo = signal->vLongEgo;
in_notifyCollision = signal->notifyCollision;
in_freqLateralOscillation = signal->freqLate
                                          on;
in_zeta = signal->zeta;
in_LatDisplacement = signal->LatDisplacemer
in_GainDeltaPhi = signal->GainDeltaPhi;
in_HeadingError = signal->DeltaPhi;
```

Modular driver-architecture

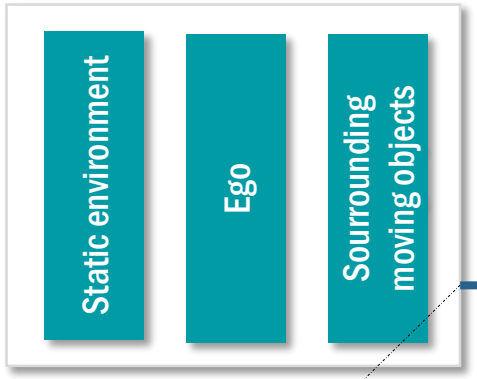
Signal classes



...

Modular driver-architecture

Signal classes



information acquisition

New signal classes integrated

- ALongitudinalDriverToDLongitudinalSignal.h
- brakingSignal.h
- carInfo.h
- commonTools.h
- complexsignals.h
- DLateralToPRFSSignal.h
- egodata.h
- globalDefinitions.h
- lateralDriverSignal.h
- primitiveSignals.h
- secondaryDriverTasksSignal.h
- staticenvironmentdata.h
- steeringSignal.h
- surroundingmovingobjectsdata.h
- utilities.h
- vector2d.h
- vector3d.h
- vectorSignals.h

```
const std::shared_ptr<structSignal<std::vector<StaticObjectsData>> const> signal = std::dynamic_pointer_cast<structSignal<std::vector<StaticObjectsData>> const>(data);
if (!signal)
{
    const std::string msg = COMPONENTNAME + " invalid signaltype";
    LOG(CbkLogLevel::Debug, msg);
    throw std::runtime_error(msg);
}
IA_I_BU.StaticObjects = signal->value;
```

Modular driver-architecture

Implementation in openPASS

Algorithm_ModularDriverArchitecture_implementation

... .cpp

Update_Input(...)

IA_Input_BU.VisualObjectsContainer = signal -> value

Update_Output(...)

Trigger(...)

InformationAcquisition_Output *IA_Output

Information_Acquisition(&IA_Input_TD, &IA_Input_BU, IA_Output)

Information_Acquisition(*Input_TD, *Input_BU, *&Output)

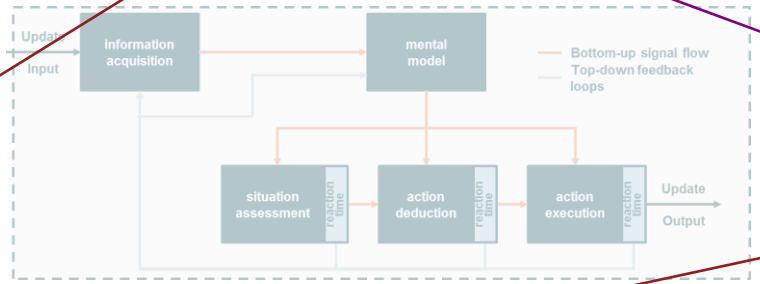
Informationacquisition.Set_Input(Input_TD, Input_BU)

Informationacquisition.Do_Function()

Output = Informationacquisition.Get_Output(...)

... .h

InformationAcquisition informationacquisition;



ContainerStructures

struct InformationAcquisition_Input_BU
{...};
...

AlgorithmInformationAcquisition

class *InformationAcquisition*

InformationAcquisition_Input_BU *IA_Input_BU;
InformationAcquisition_Input_TD *IA_Input_TD;
InformationAcquisition_Output IA_Output;

Set_Input(*Input...)

Do_Function()

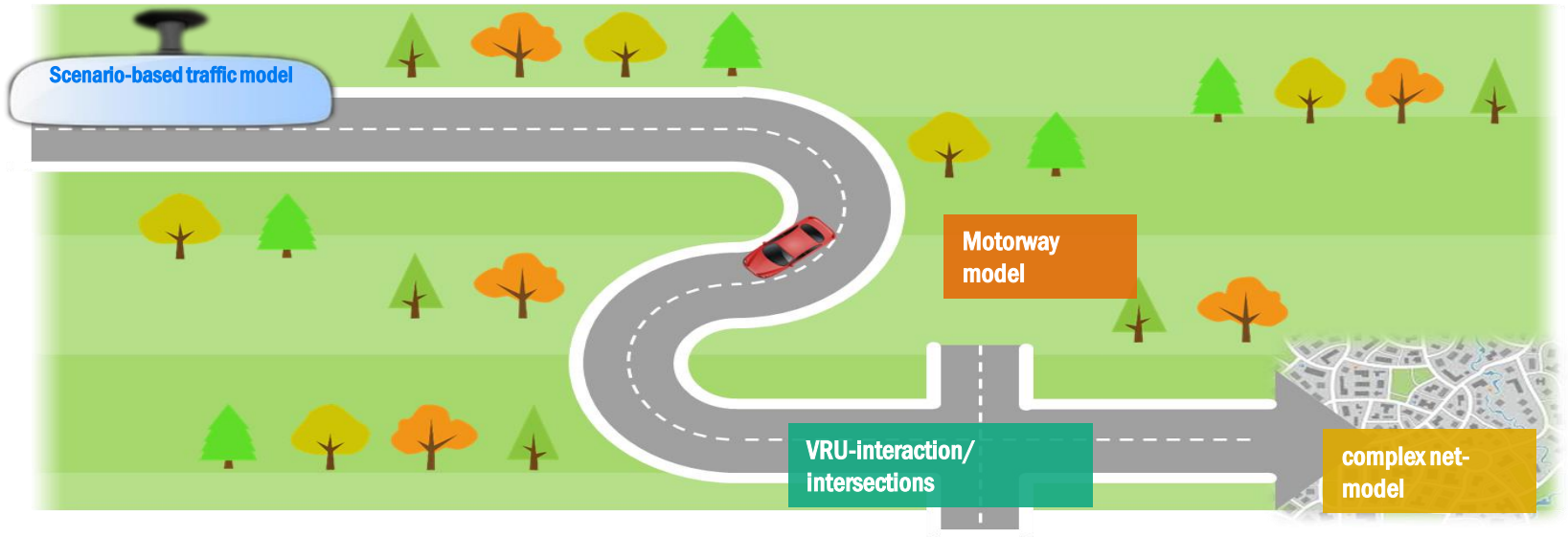
InformationAcquisition_Output *Get_Output()

Agenda

1. Modular driver-architecture
2. Usecase and dependencies on the framework

Usecase and dependencies on the framework

Our Goals



Baseline-simulation of current traffic and accident events

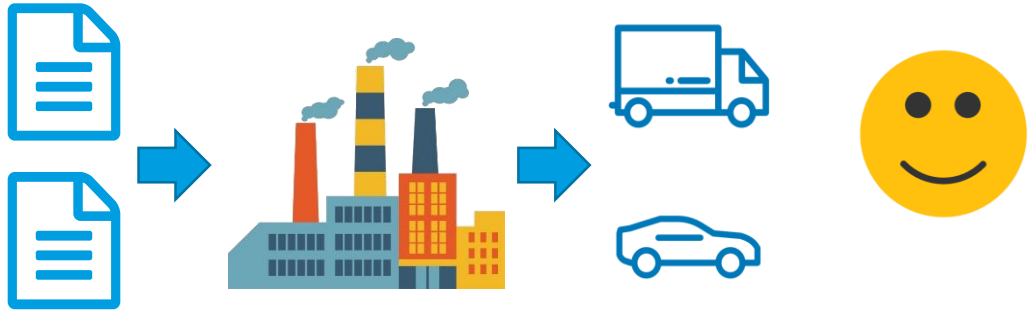
e.g. HAD valuation

Evaluationline-simulation

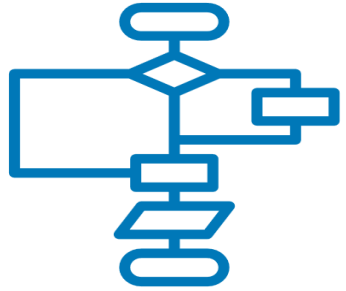
e.g. Evaluation of the influence of highly automated vehicles on traffic safety

Usecase and dependencies on the framework

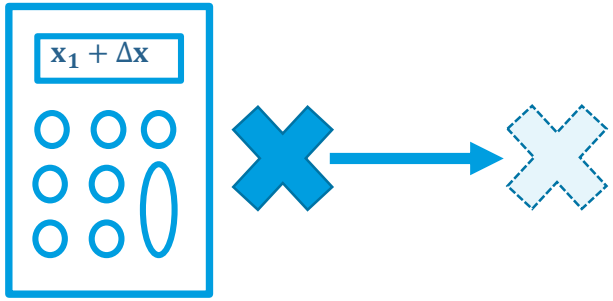
What does the framework have to provide?



1. Agent factory



3. Program flow



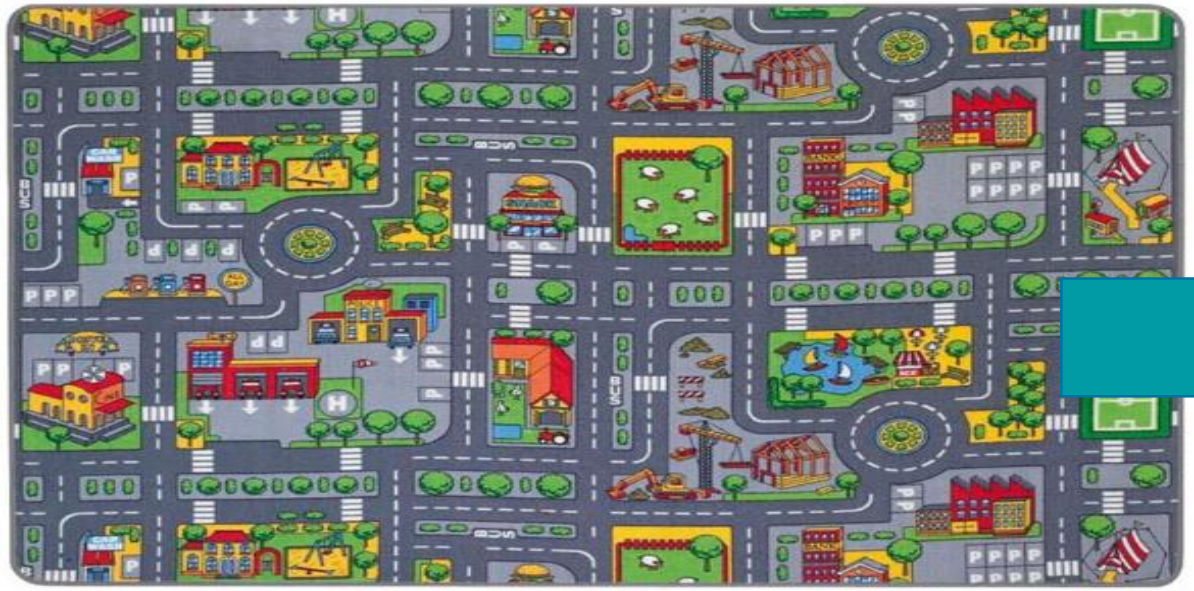
2. Kinematic calculation/
Coordinate axes

4. Infrastructure

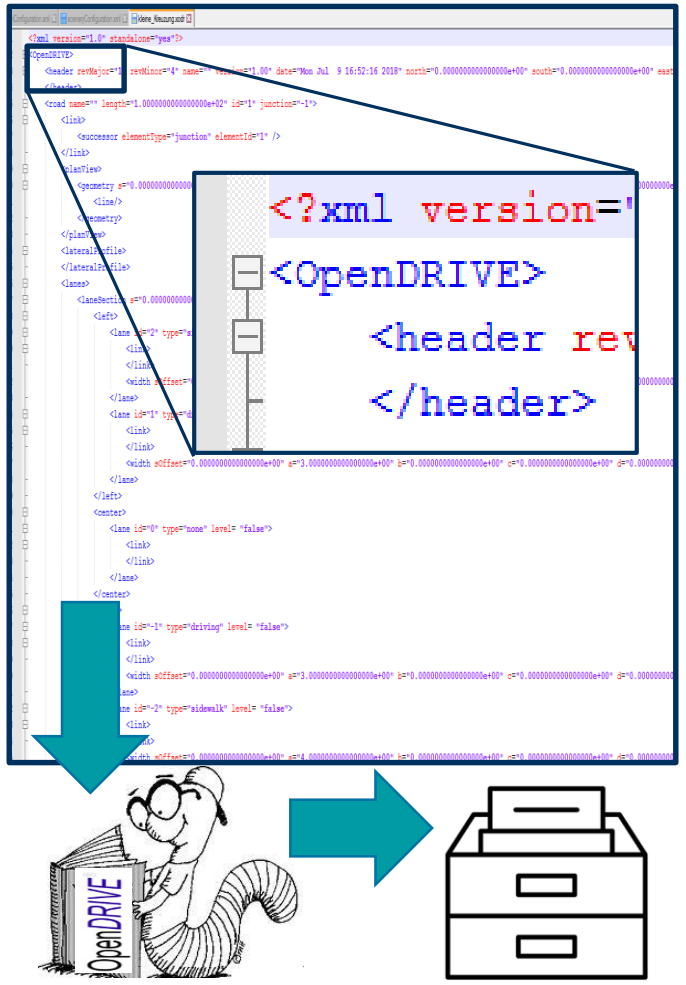
Bildquellen: http://getdrawings.com/vector-tag/factory; https://www.flaticon.com/free-icon/delivery-truck_259569; https://thenounproject.com

Infrastructure

User playground

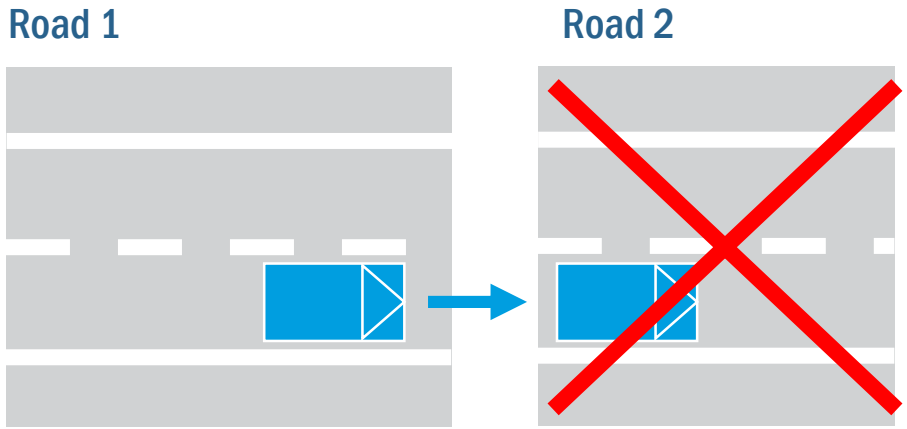


OpenDRIVE



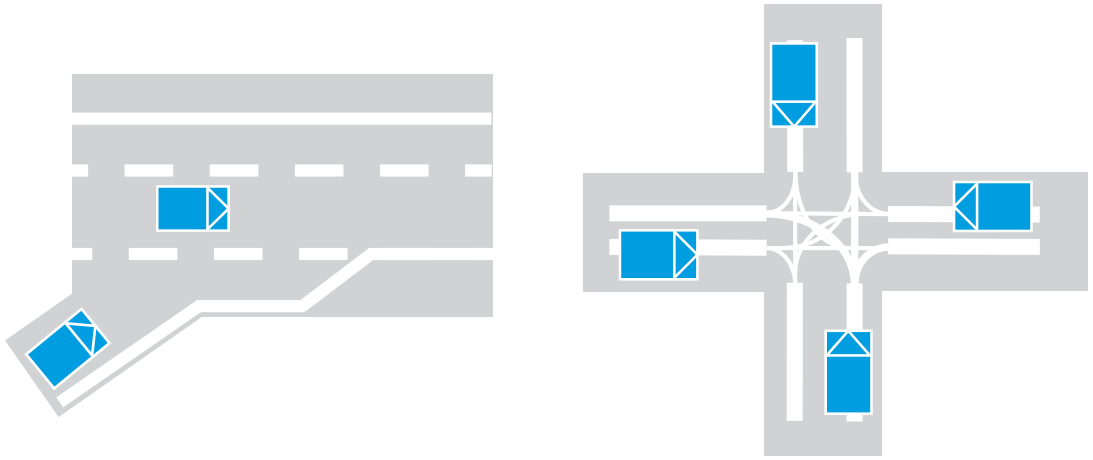
Bildquellen: https://www.diejuniorkiste.de

Current state



- Framework can only process a single road
- It is not possible to connect different roads as at intersections

What we want to do



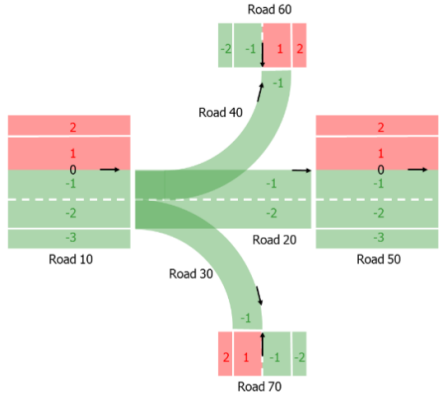
- Parsing and instantiation of a complete road network
- Detecting where we are and where we can navigate to

→ Extension of the World- and AgentInterface

Bildquellen: https://wallsheaven.de

Infrastructure
underlying problems


```
void BaseTrafficObjectLocator::Locate()
{
  // NOTE: assume single road in world for now
  auto sections = roads.begin()->second->GetSections();
```



Expand parsing of OpenDRIVE (Intersections)

Expand instantiation and access on instantiated objects

Vehicle can only locate on the first road in roads-container.
Solution: → Selection of the current road via AgentAdapter

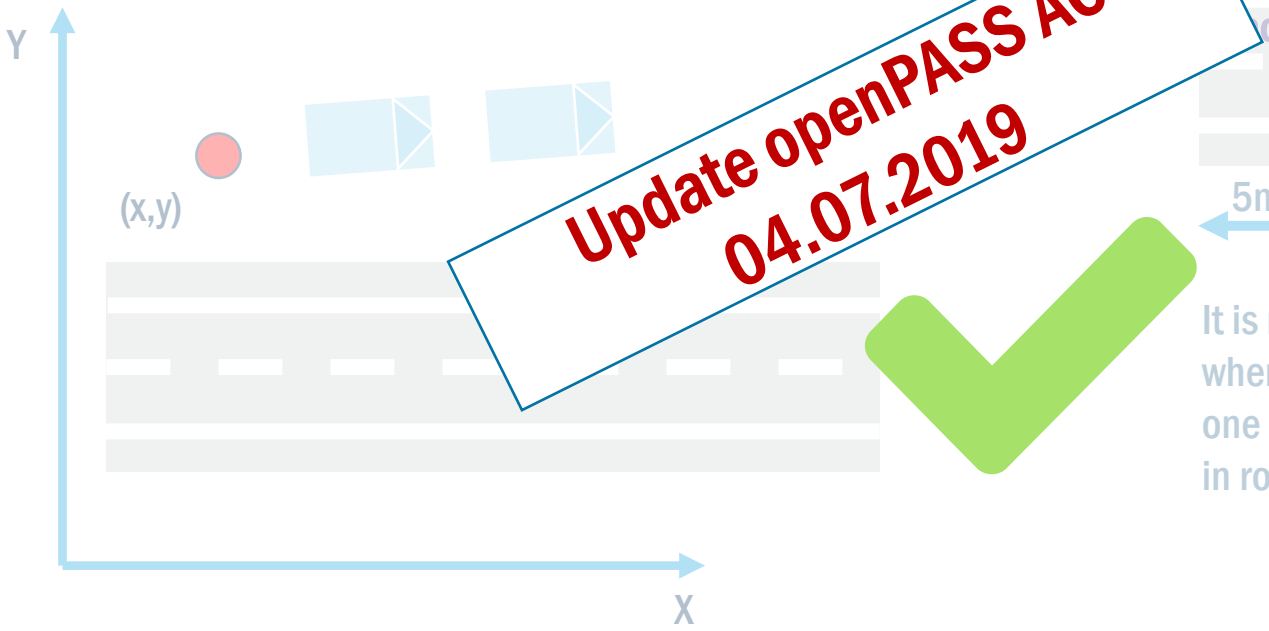
Expand infrastructure:
- Create a new class Intersection
- Expand existing Lane/Road classes

Bildquellen: https://www.diejuniorkiste.de

Suggestions for Improvement

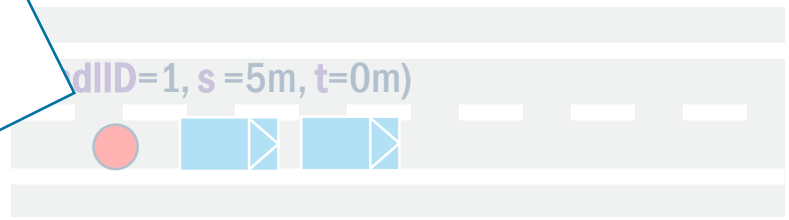
Spawn point

Current state:
Spawn point is given by global coordinates (x,y).
If Spawn point is not located on a road
→spawned cars will be deleted

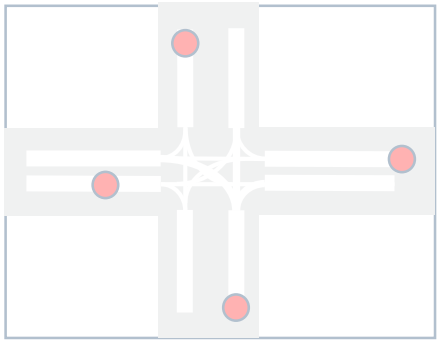


Update openPASS AC
04.07.2019

Suggestions :
define spawn point by chosen
roadID/laneID and road coordinate s, t



It is much more fail-safe
when more than
one spawn is used
in road networks.



Suggestions for Improvement

Kinematic calculation AgentAdapter

Example methods with unclear or incomplete behavior

```
double GetAccelerationY() const override
{
    return 0.0;
}
```

```
double GetVelocityY() const override
{
    return 0.0;
}
```

```
double GetVelocityLateral() override
{
    // TODO
    return 0.0;
}
```

GetVelocityLongitudinal is missing !

```
void UpdateVelocityY(double velocityY)
{
    this->velocityY = velocityY;
    /* NOTE: not used now
       TODO: map to OSI
    OWL::Primitive::AbsVelocity velocity = baseTrafficObject->GetAbsVelocity();
    velocity.vx = ...
    velocity.vy = velocityY;
    baseTrafficObject->SetAbsVelocity(velocity);
    */
}
```

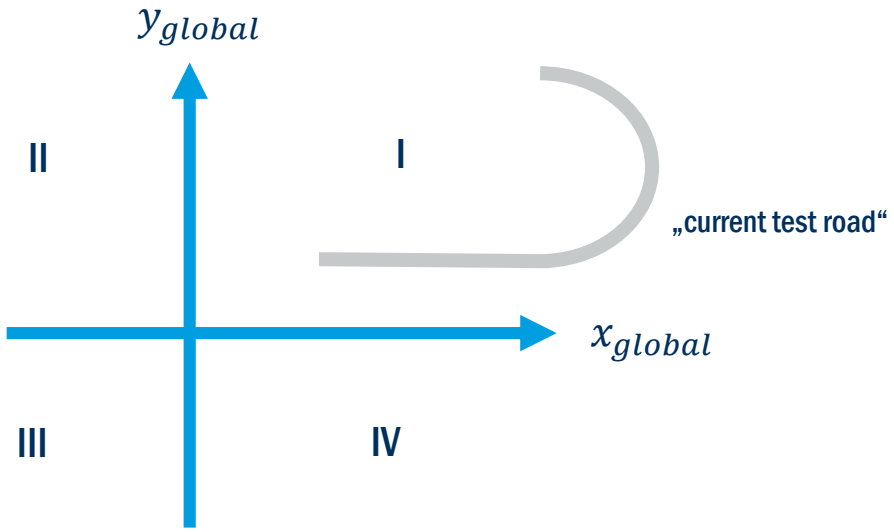
```
void UpdateAccelerationY(double accelerationY)
{
    this->accelerationY = accelerationY;
    /* NOTE: not used now
       TODO: map to OSI
    OWL::Primitive::AbsAcceleration acceleration = baseTrafficObject->GetAbsAcceleration();
    acceleration.ay = accelerationY;
    baseTrafficObject->SetAbsAcceleration(acceleration);
    */
}
```

Suggestions for Improvement

Kinematic calculation AgentAdapter

- Naming of VelocityX and VelocityY is confusing:
  →Sounds like it refers to global coordinates, but refers to Agent-coordinates

- Doubt whether the calculation of VelocityX is correct for each quadrants of the coordinate system.



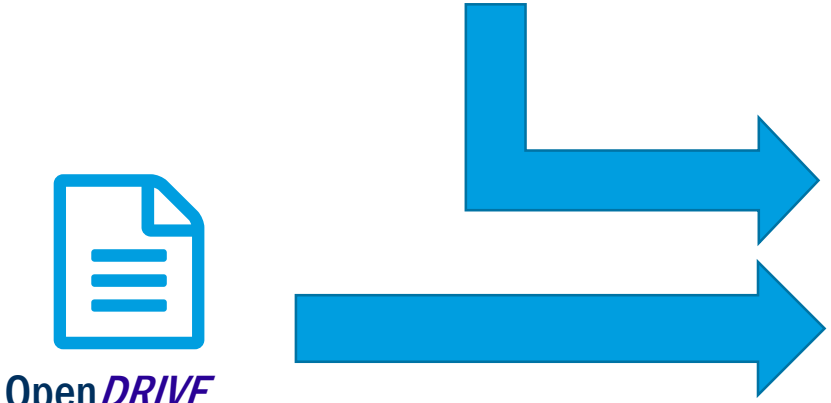
Wish from the user's point of view:

1. Implement the TODOs
2. Verification of all kinematic calculation for all quadrants
3. Reconsider naming

VERY nice to have...

Visualization of the output

```
<Sample Time="0">0.000000, 0, 80.000000, 15.613975, -1.499974, 128.000000, -1.570796, 0.000000, 0, 70.000000, 11.514026, 35.999999, 105.499976, 3.141593</Sample>
<Sample Time="100">0.000000, 1, 81.561397, 15.613975, -1.499973, 126.438602, -1.570796, 0.000000, 1, 71.151403, 11.514026, 34.848597, 105.499975, 3.141593</Sample>
<Sample Time="200">9.810000, 1, 83.024695, 14.632975, -1.499973, 124.975305, -1.570796, 9.810000, 1, 72.204705, 10.533026, 33.795294, 105.499975, 3.141593</Sample>
<Sample Time="300">9.810000, 1, 84.389892, 13.651975, -1.499972, 123.610107, -1.570796, 9.810000, 1, 73.159908, 9.552026, 32.840092, 105.499975, 3.141593</Sample>
<Sample Time="400">9.810000, 1, 85.656990, 12.670975, -1.499972, 122.343010, -1.570796, 9.810000, 1, 74.017010, 8.571026, 31.982989, 105.499974, 3.141593</Sample>
<Sample Time="500">9.810000, 1, 86.825987, 11.689975, -1.499972, 121.174012, -1.570796, 9.810000, 1, 74.776013, 7.590026, 31.223986, 105.499974, 3.141593</Sample>
<Sample Time="600">9.810000, 1, 87.896885, 10.708975, -1.499971, 120.103115, -1.570796, 9.810000, 1, 75.436916, 6.609026, 30.563084, 105.499974, 3.141593</Sample>
<Sample Time="700">9.810000, 1, 88.869682, 9.727975, -1.499971, 119.130317, -1.570796, 9.810000, 1, 75.999718, 5.628026, 30.000281, 105.499974, 3.141593</Sample>
<Sample Time="800">9.810000, 1, 89.744380, 8.746975, -1.499971, 118.255620, -1.570796, 9.810000, 1, 76.464421, 4.647026, 29.535579, 105.499974, 3.141593</Sample>
<Sample Time="900">9.810000, 1, 90.520977, 7.765975, -1.499970, 117.479022, -1.570796, 9.810000, 1, 76.831024, 3.666026, 29.168976, 105.499973, 3.141593</Sample>
<Sample Time="1000">9.810000, 1, 91.199475, 6.784975, -1.499970, 116.800525, -1.570796, 9.810000, 1, 77.099526, 2.685026, 28.900473, 105.499973, 3.141593</Sample>
<Sample Time="1100">9.810000, 1, 91.779872, 5.803975, -1.499970, 116.220127, -1.570796, 0.019218, 0, 77.367837, 2.683104, 28.632163, 105.499973, 3.141593</Sample>
<Sample Time="1200">9.810000, 1, 92.262170, 4.822975, -1.499970, 115.737830, -1.570796, 0.019218, 0, 77.635955, 2.681183, 28.364045, 105.499973, 3.141593</Sample>
<Sample Time="1300">9.810000, 1, 92.646367, 3.841975, -1.499970, 115.353632, -1.570796, 0.019218, 0, 77.903881, 2.679261, 28.096119, 105.499973, 3.141593</Sample>
<Sample Time="1400">9.810000, 1, 92.932465, 2.860975, -1.499970, 115.067535, -1.570796, 0.019218, 0, 78.171615, 2.677339, 27.828385, 105.499973, 3.141593</Sample>
```



E.g. Covise

Bildquellen: https://www.v-kon.media/produkte_verkehrssimulation.php

Thank you for your attention!