Universität Augsburg
Fakultät für Angewandte Informatik

Praktikum Simulation von autonomen Fahrzeugen

Vorstellung 31.03.21

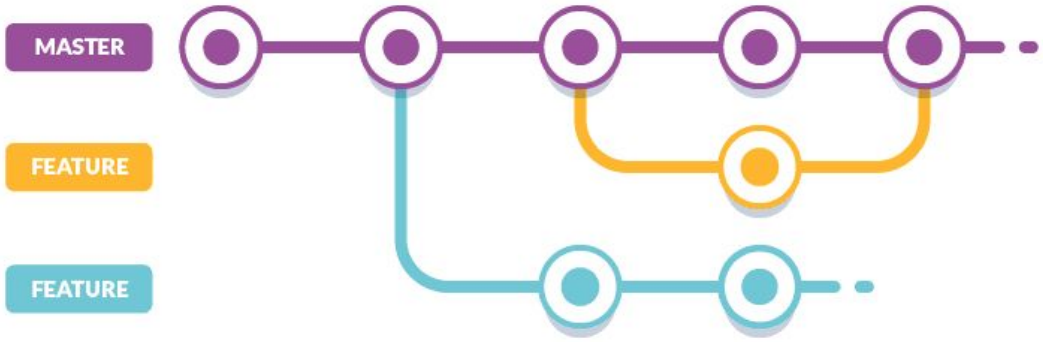
Gruppe 2

Julius Brandl, Wolfgang Lang, Lukas Hartmann, Maurizio Volanti, Yulia Khlyaka, Valentin Höpfner

Content

1. Workflow
2. Global Planning
3. Cruising
   a. Steering Controller
   b. Radar
4. Behavior Tree
5. Overtaking
   a. Lidar
   b. Maneuvers
6. Intersection
   a. Stop Line detection
   b. Street Object Detection
7. Conclusion

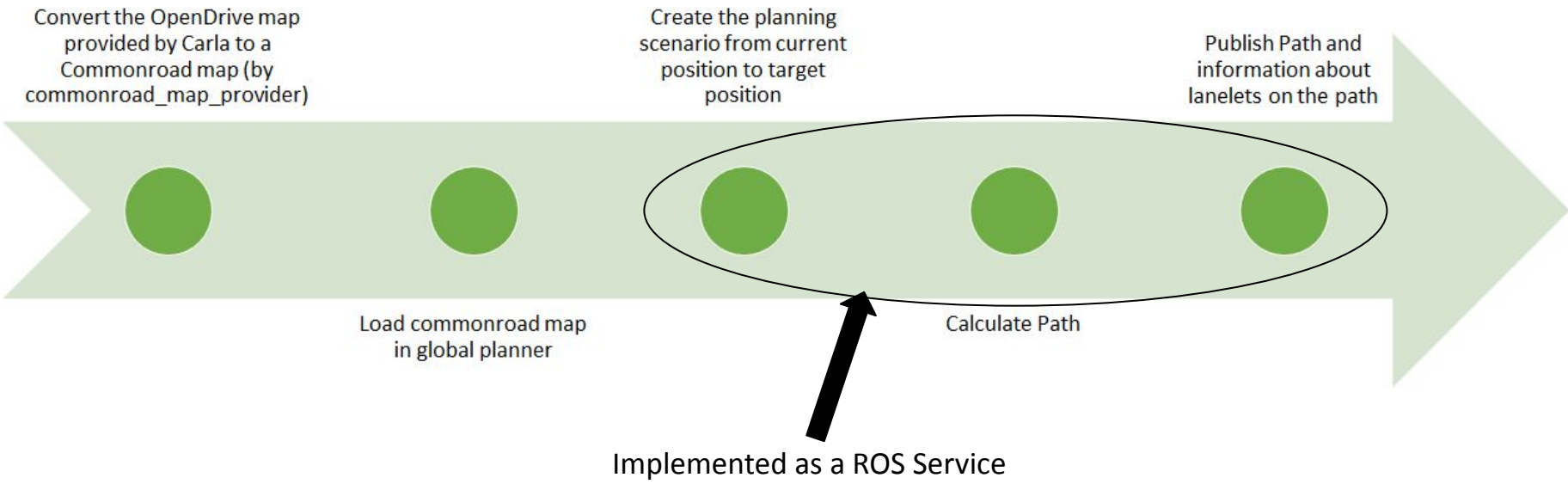
Workflow



- Feature description and assignment with Issues
- Master is protected
- Merging to master is only possible if pull request is approved by at least one
- Pull request should be checked by someone who was not involved at developing the feature

Global Planner

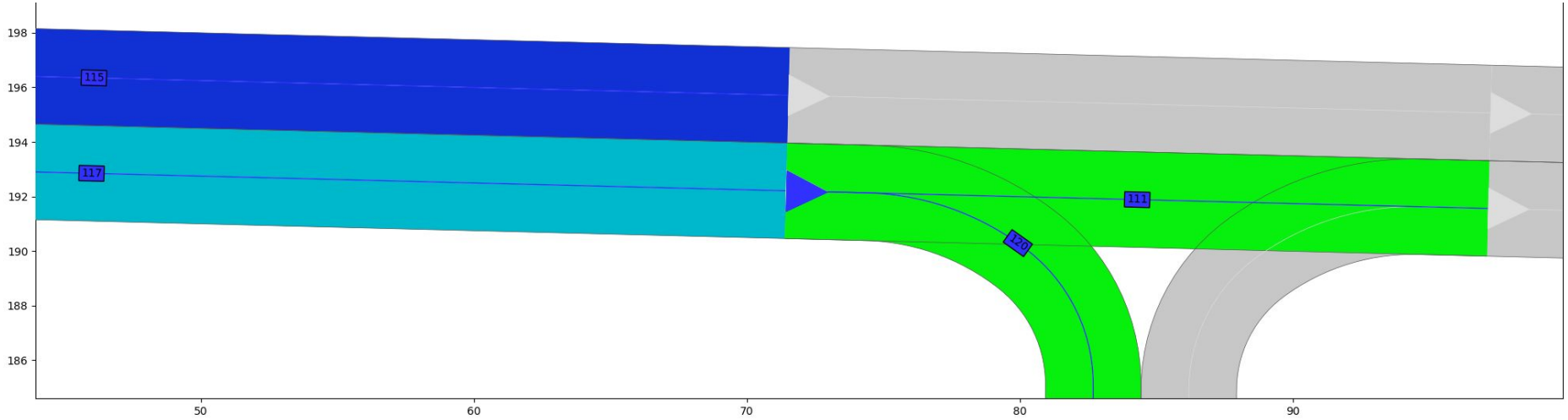
Universität Augsburg
Fakultät für Angewandte Informatik



Global Planner

Universität Augsburg
Fakultät für Angewandte Informatik

Lanelets

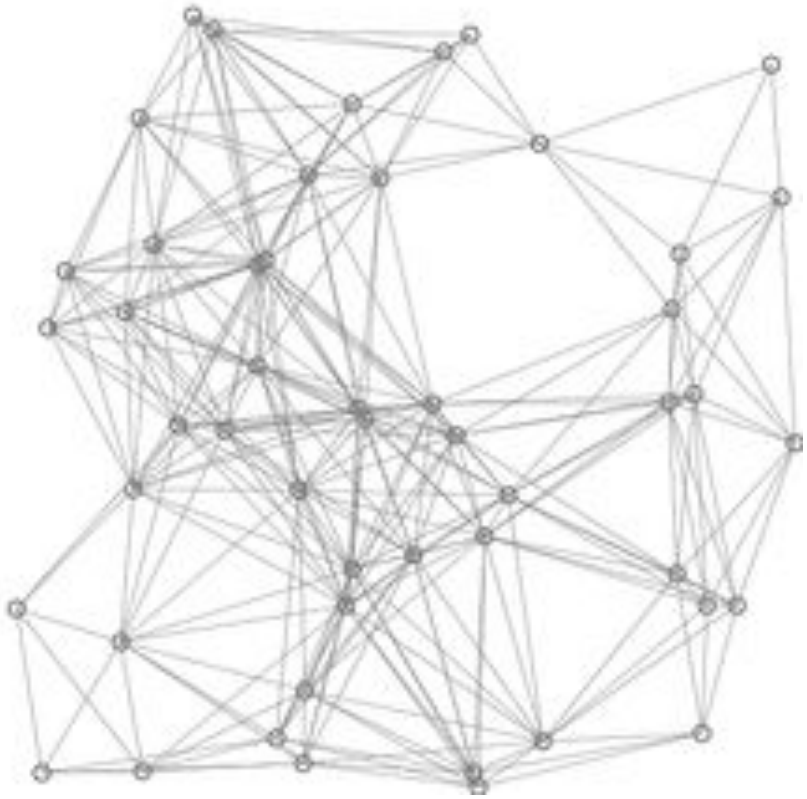


- Cuts road network in separated sections (lanelet).
- Every lanelet owns a left, right and centerline. Those lines are described by polylines.
- Every lanelet can own a left/right adjacent lanelet.
- Every lanelet can own zero to n predecessors/successors.
- Those lanelets form a graph.

Global Planner

Universität Augsburg
Fakultät für Angewandte Informatik

- Planning is done via graph search (Dijkstra) from Commonroad.
- Publish the path as array of points.
- Publish information that is used for the local planner:
  - lanelets on the global path.
  - lanelets that form parallel lanes.
  - lanelets that belong to an intersection.



Core functionalities

If we don’t consider traffic rules or more complex behaviours, the aim is to compute Vehicle Control Commands, so that

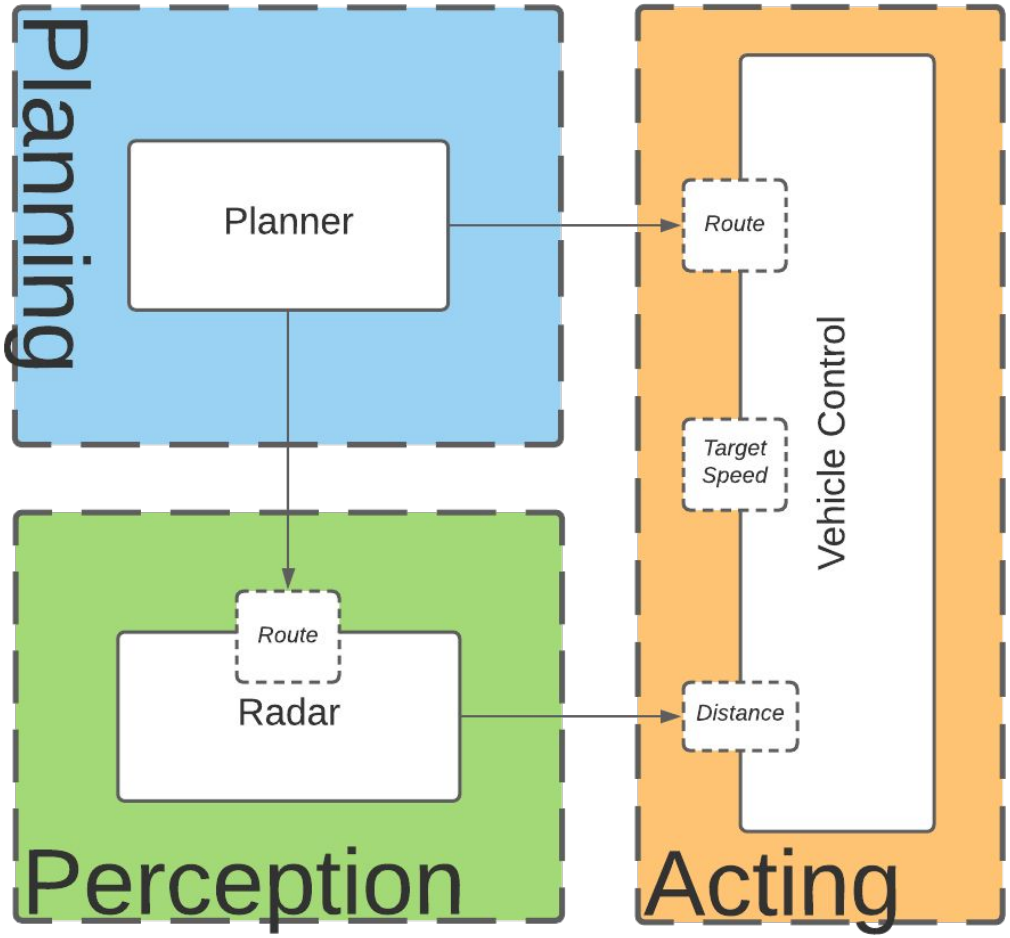
- we get from A to B,
- at a controlled speed
- without crashing into anything.

To achieve this, we need

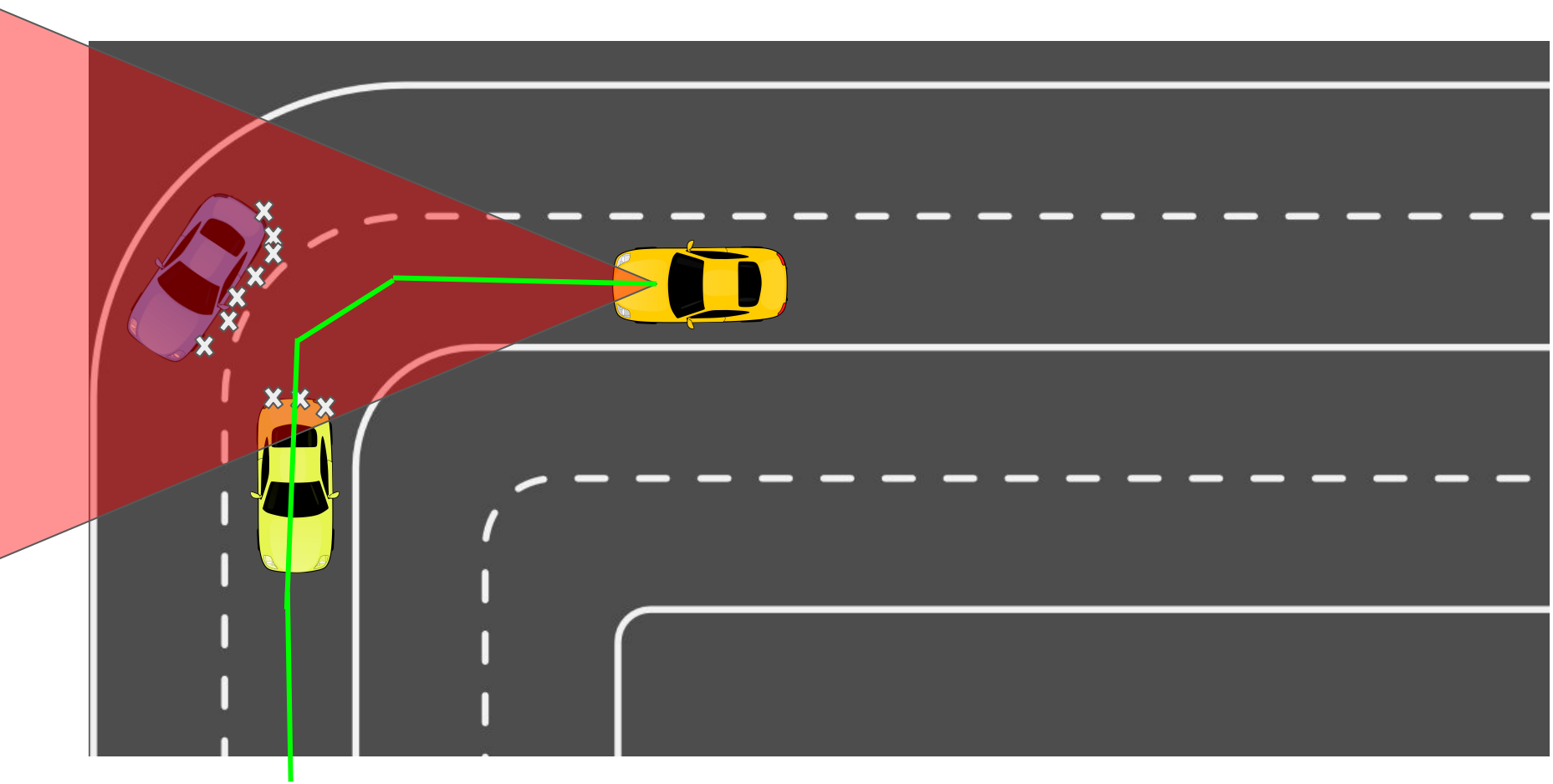
- a path from A to B, that avoids static obstacles,
- a sensor to detect dynamic obstacles on this path,
- a set of controllers that compute steering commands from this information.

Core functionalities as ROS-Nodes

Universität Augsburg
Fakultät für Angewandte Informatik

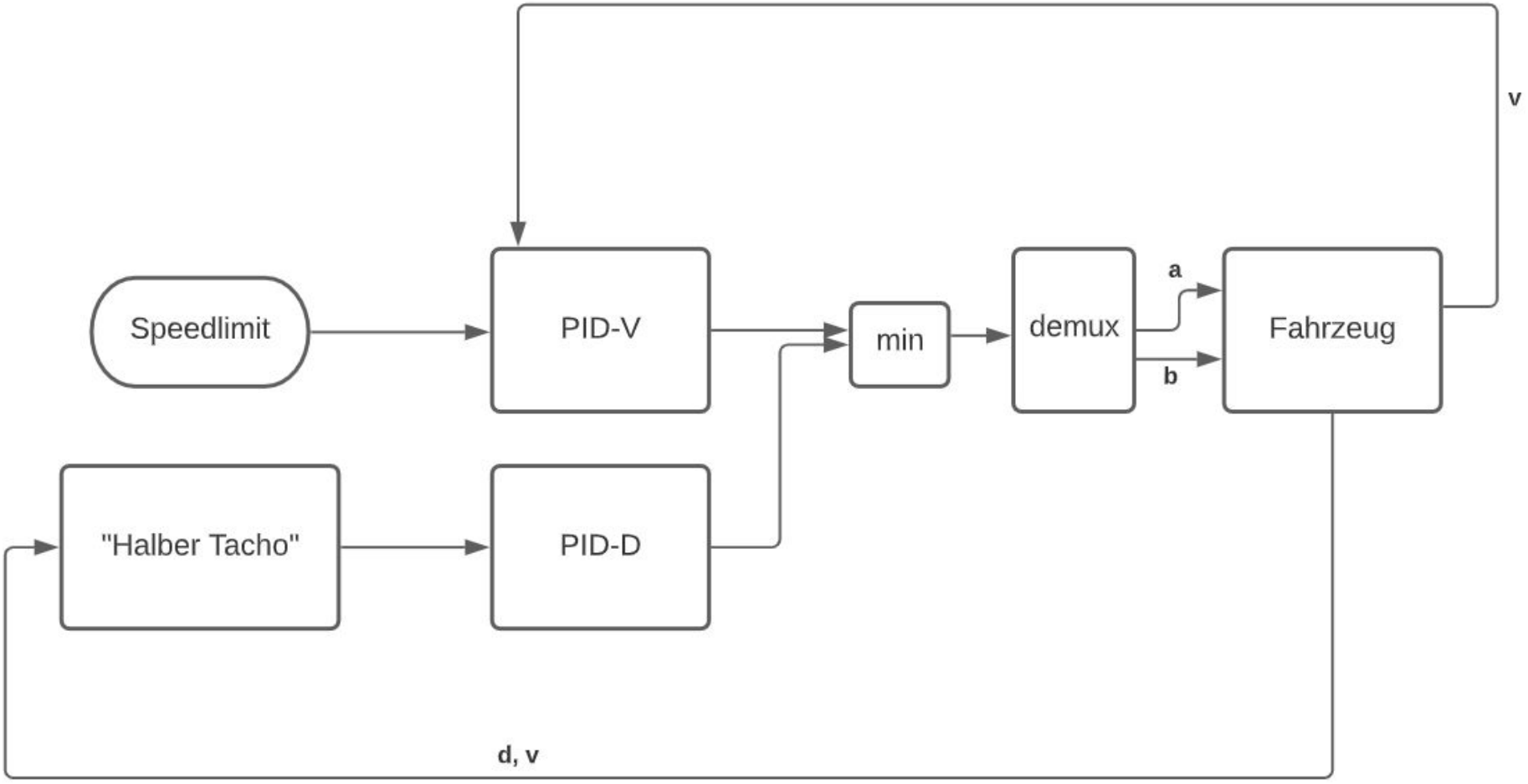


Radar



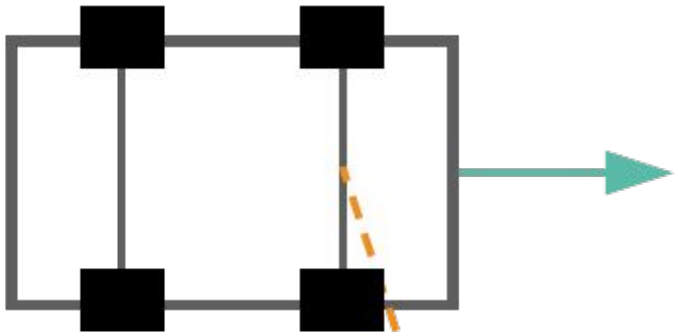
Vehicle Control - Longitudinal

Universität Augsburg
Fakultät für Angewandte Informatik

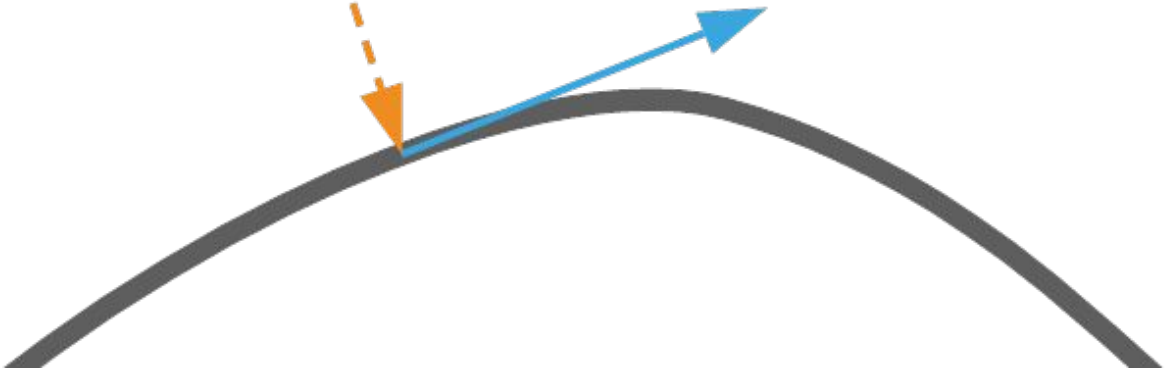


Vehicle Control - Lateral

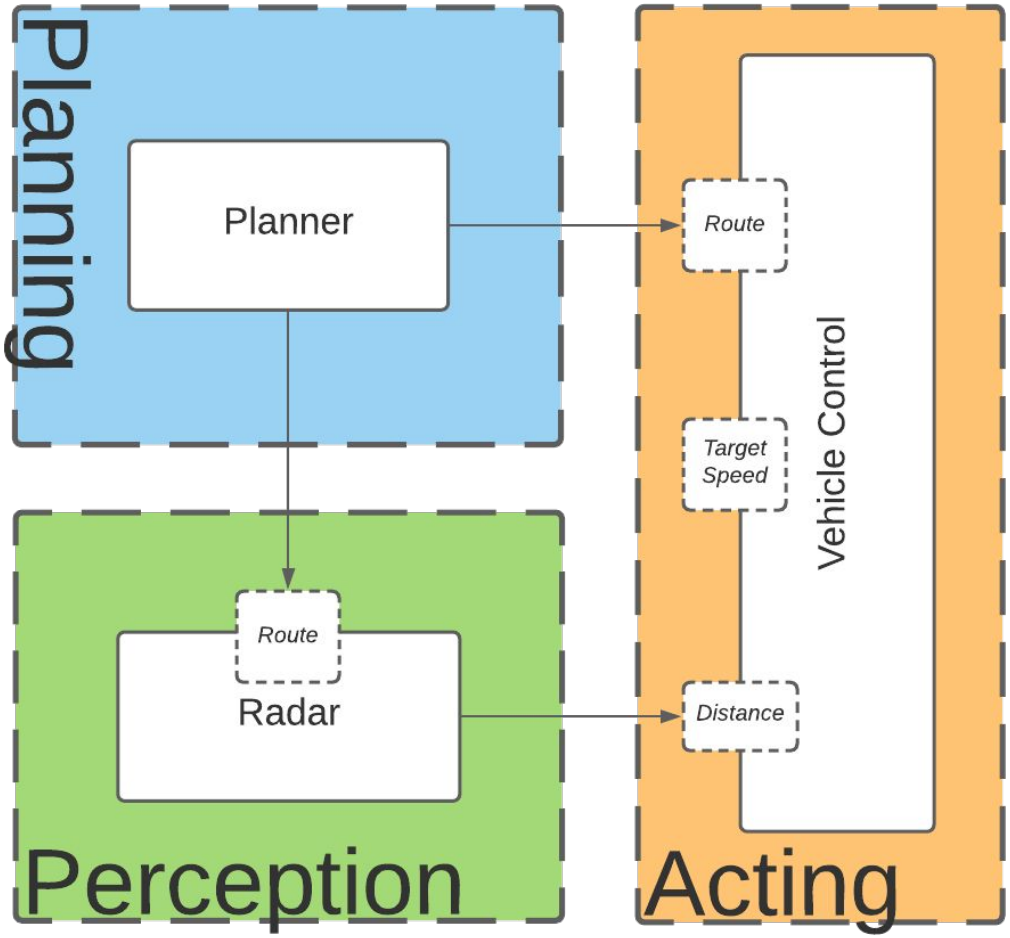
Universität Augsburg
Fakultät für Angewandte Informatik



$$s = \theta_e + \theta_d$$

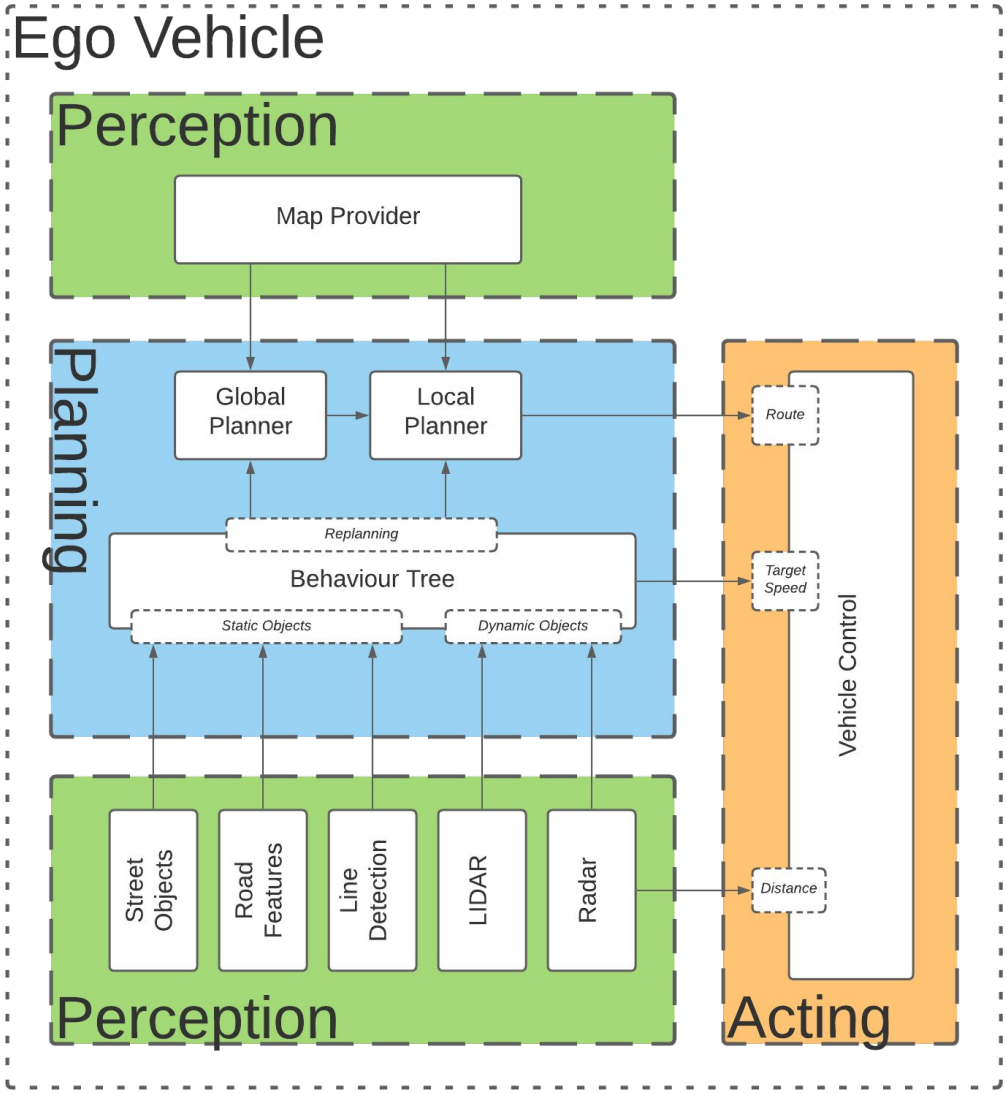


Basic Nodes


Planning
Planner
Route
Target Speed
Distance
Vehicle Control
Route
Radar
Perception
Acting

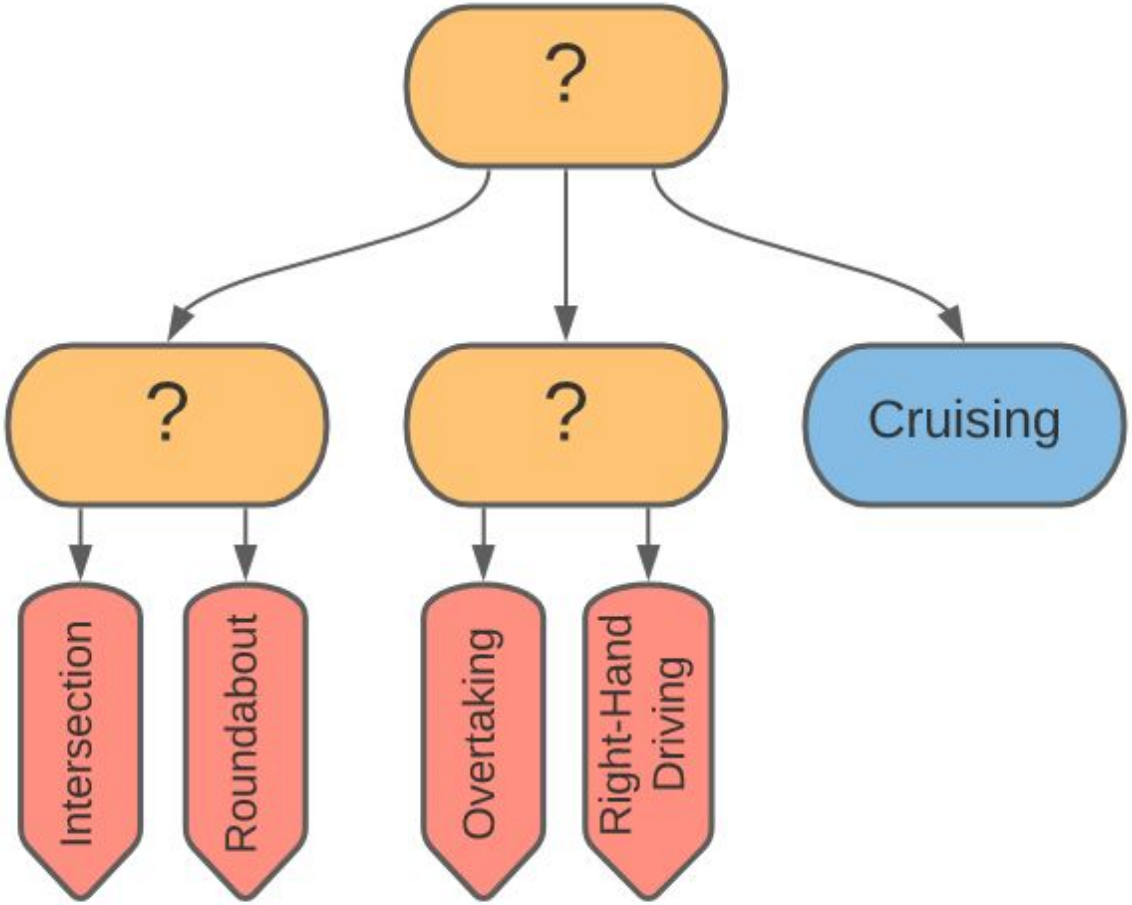
All Nodes

Universität Augsburg
Fakultät für Angewandte Informatik



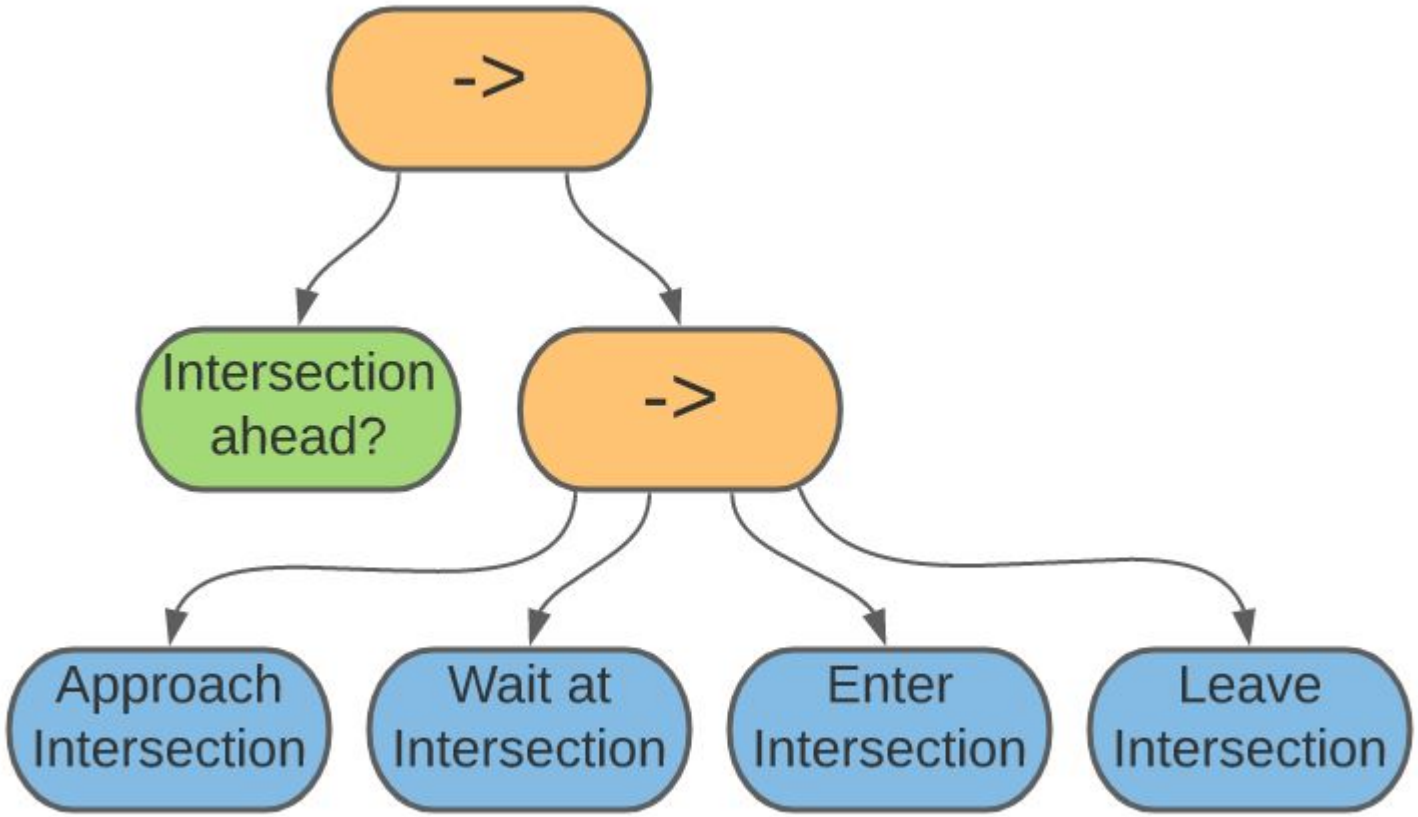
Our Behavior Tree - Overview

Universität Augsburg
Fakultät für Angewandte Informatik



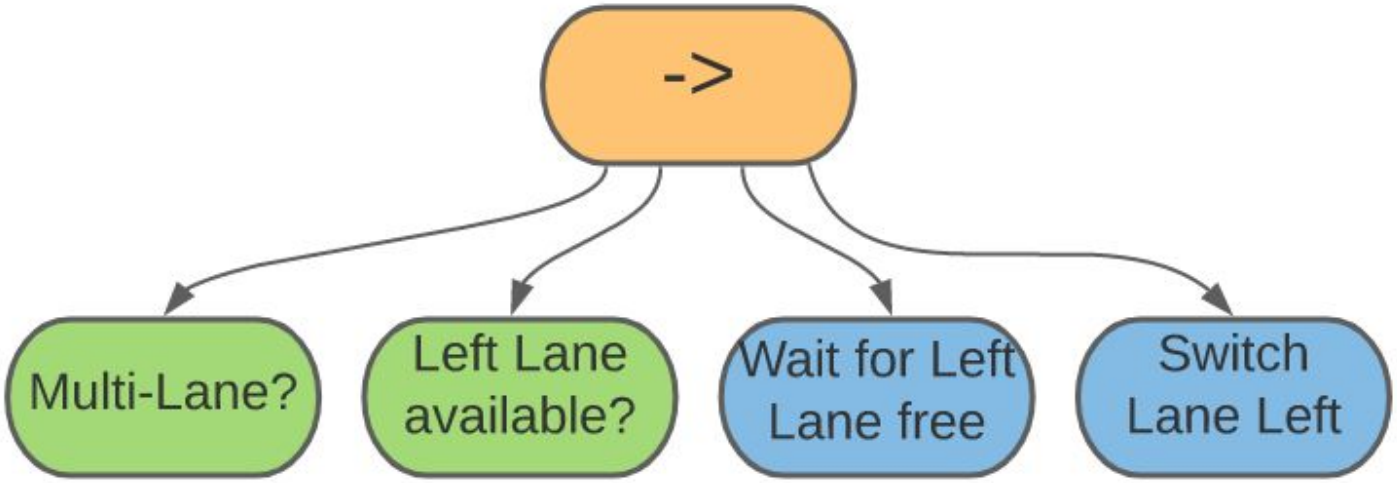
Our Behavior Tree - Intersection Subtree

Universität Augsburg
Fakultät für Angewandte Informatik



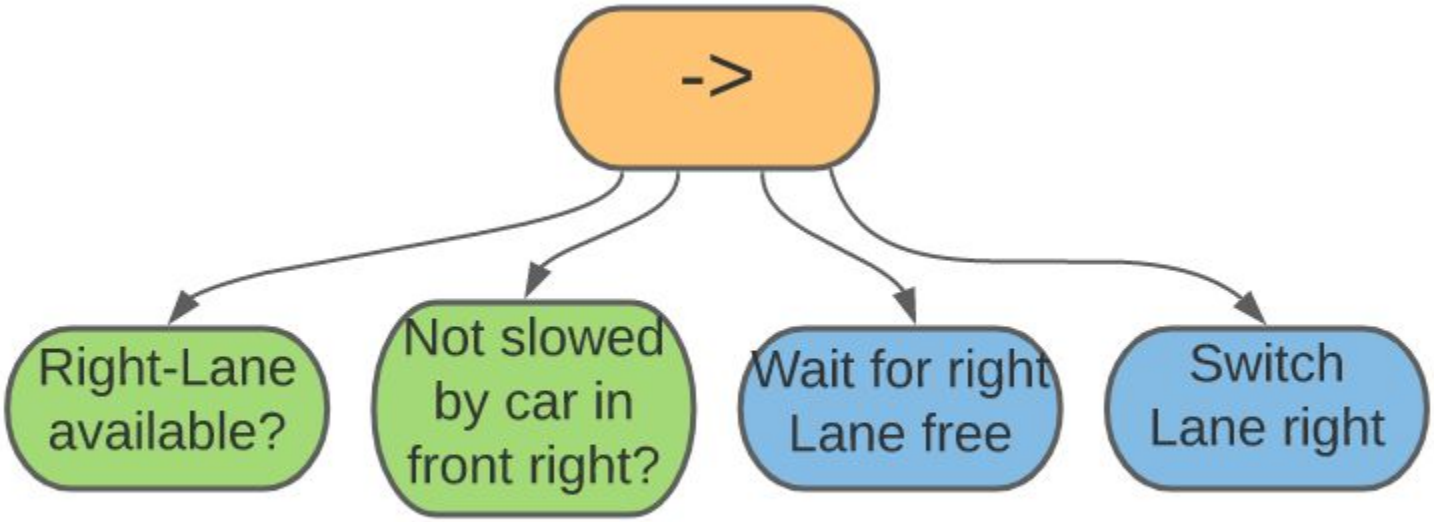
Our Behavior Tree - Overtaking

Universität Augsburg
Fakultät für Angewandte Informatik



Our Behavior Tree - Right-Hand Driving

Universität Augsburg
Fakultät für Angewandte Informatik



Behavior Tree - Strengths

Universität Augsburg
Fakultät für Angewandte Informatik

- Easy to modify/extend
- Easy to understand/read
- Modularity
- Reactivity
- Can be visualized easily (useful for debugging)
- Whole skeleton can be implemented in advance -> not everyone needs to know how BT works

Overtaking - Lidar

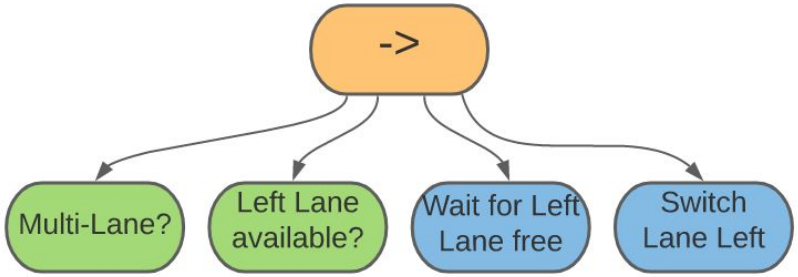
Universität Augsburg
Fakultät für Angewandte Informatik

Main Function:

Overtaking - Lidar

Universität Augsburg
Fakultät für Angewandte Informatik

Main Function:



Overtaking - Lidar

Universität Augsburg
Fakultät für Angewandte Informatik

Adjustability:

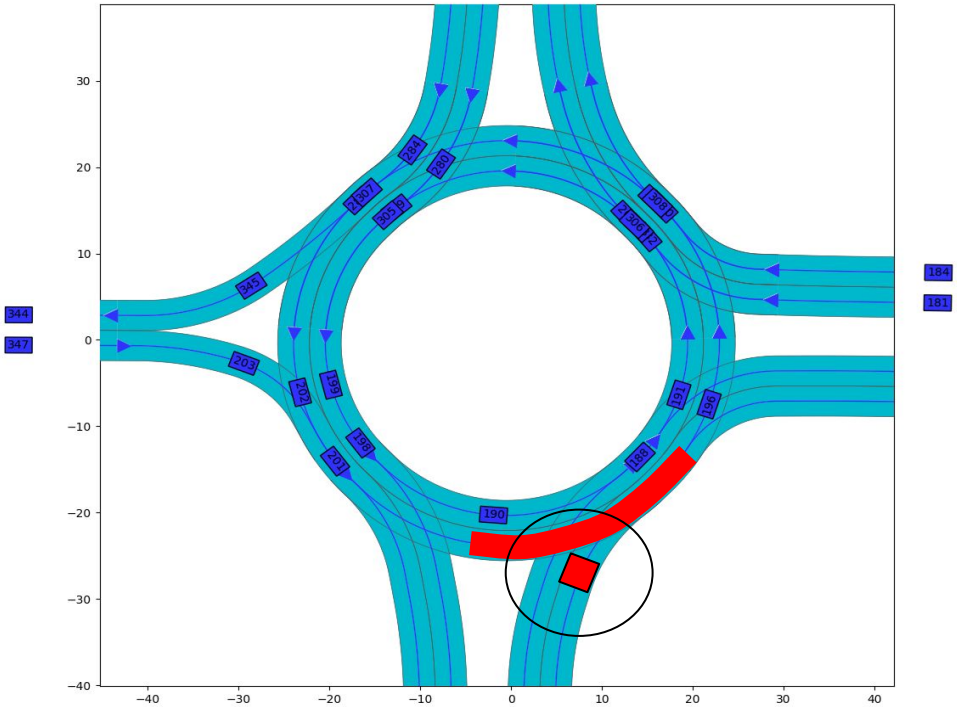
- **amount of points**
- field of view
- position of sensor
- range

Overtaking - Lidar

Universität Augsburg
Fakultät für Angewandte Informatik

Ros-Service:

- transforms points into map coordinates
- filters the transformed points for points on given lanelet (by id)
- returns true or false, if there is an obstacle or not



Overtaking - Maneuvers

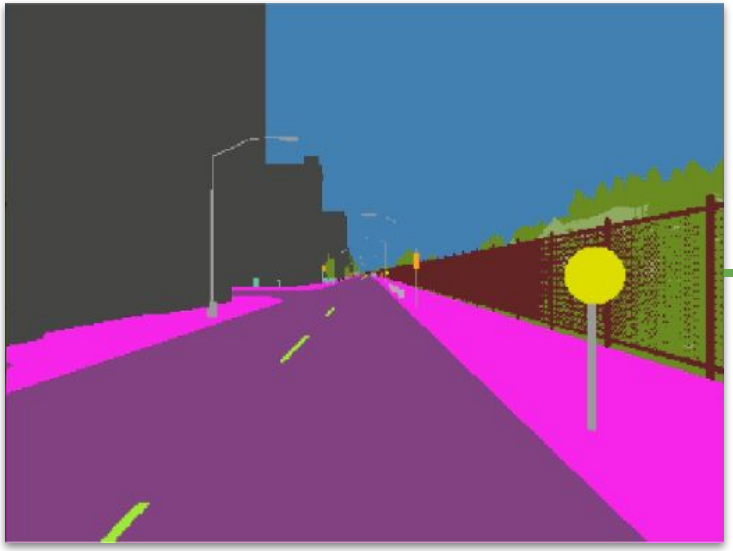
Universität Augsburg
Fakultät für Angewandte Informatik

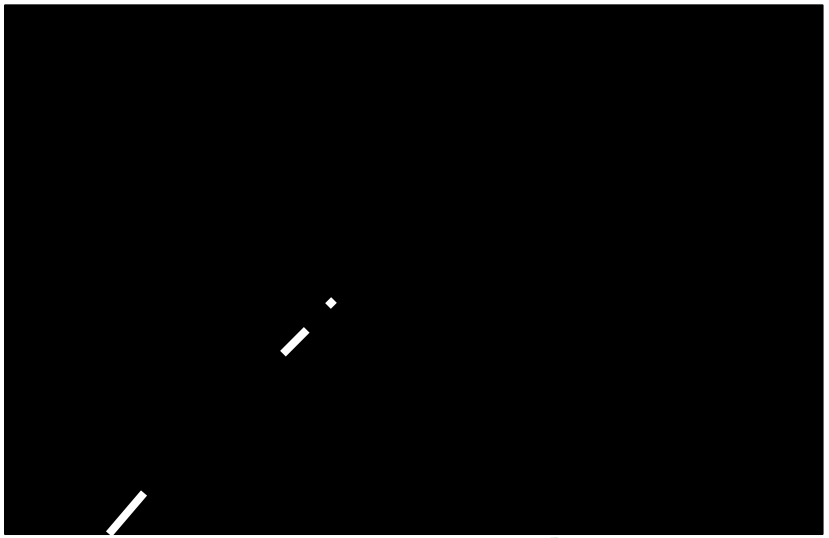
Switch-Lanes:

- called via Ros-Service “update_local_path”
- includes boolean for change to left and right lane
- next 10 points on current lane, then switch to 30th point on next lane

Intersection - Stop Line Detection

Universität Augsburg
Fakultät für Angewandte Informatik





position	angle	length
(3.2,4.8)	1.721	5.3
...	...	...

position	angle	length
(3.2,4.8)	1.721	5.3

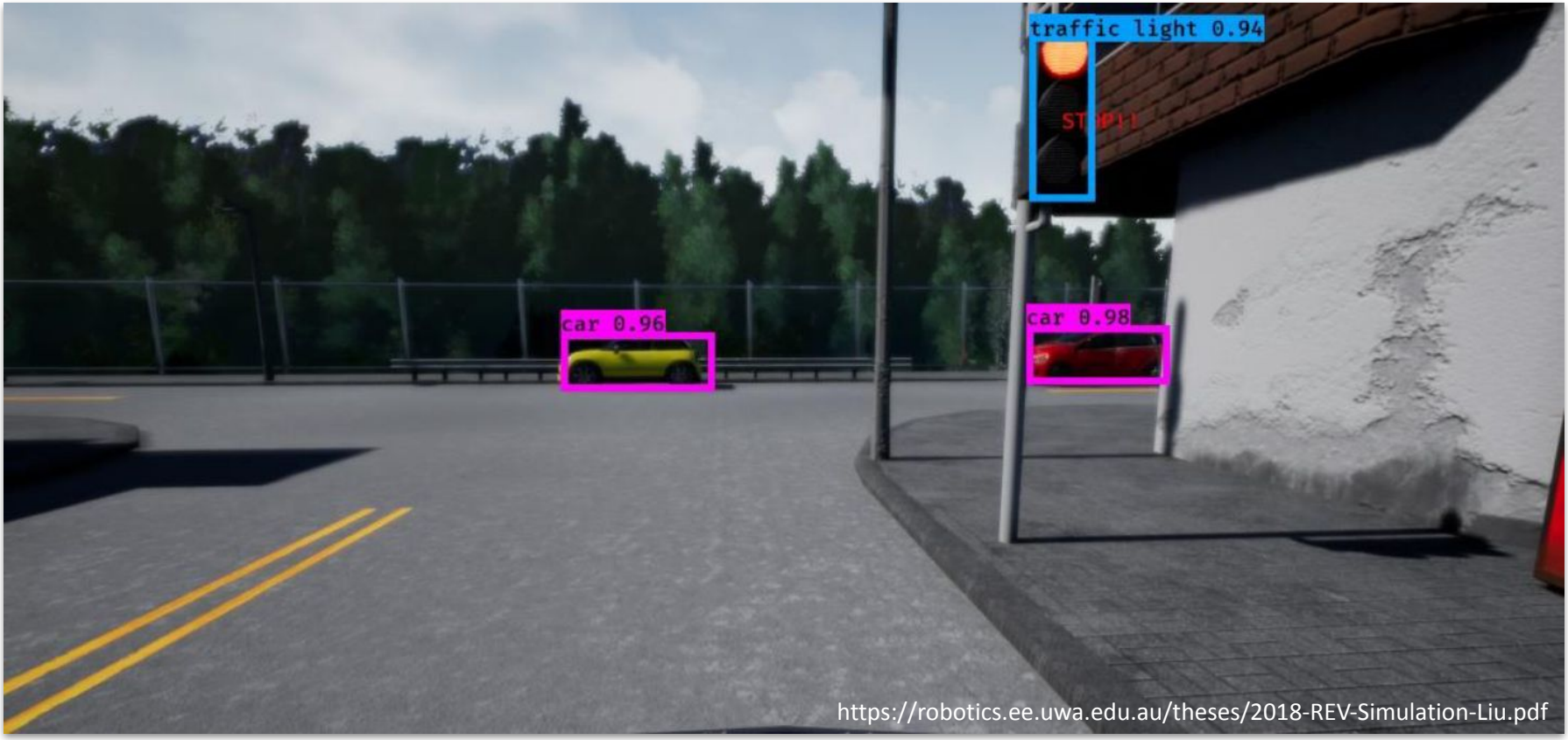
distance

Intersection - Street Object Detection

Universität Augsburg
Fakultät für Angewandte Informatik

First approach

- YOLO (You Only Look Once) real-time object detection system
- Darknet: neural network framework



Intersection - Street Object Detection

First approach

- YOLO (You Only Look Once) real-time object detection system
- Darknet: neural network framework

Disadvantages

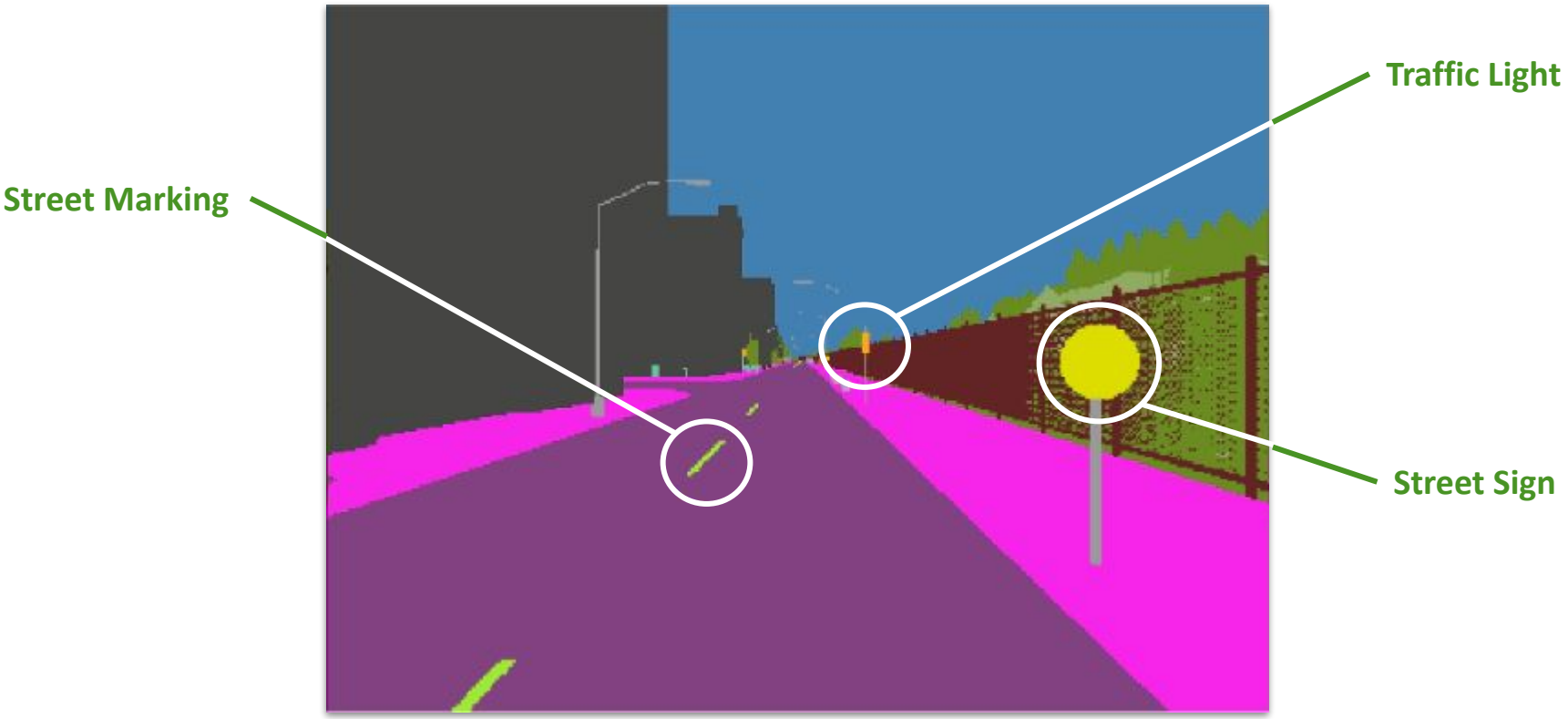
- huge dataset is needed
- needs a lot of performance
- overload for what is really needed

Intersection - Street Object Detection

Universität Augsburg
Fakultät für Angewandte Informatik

Second approach

- Semantic Segmentation Camera
- Pixel analysis
- OCR (text recognition)



Intersection - Street Object Detection

Universität Augsburg
Fakultät für Angewandte Informatik

First step (street signs)

- Cut relevant objects out of semantic segmentation camera and map them to rgb camera

Semantic Segmentation Camera

RGB Camera

Intersection - Street Object Detection

Universität Augsburg
Fakultät für Angewandte Informatik

Second step (street signs)

- Manipulate image for clear edges
- Use text recognition to get the value of speed limits

Intersection - Street Object Detection

Universität Augsburg
Fakultät für Angewandte Informatik

First step (traffic lights)

- Cut relevant objects out of semantic segmentation camera and map them to rgb camera

Semantic Segmentation Camera

RGB Camera

Intersection - Street Object Detection

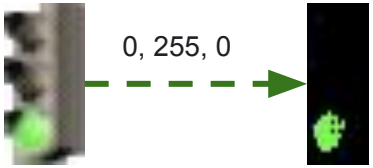
Universität Augsburg
Fakultät für Angewandte Informatik

Second step (traffic lights)

Publish in Topic → **“RED”**

Traffic lights are analyzed by the specific dominant color within the extracted image.

Publish → **“YELLOW”**



Publish in Topic → **“GREEN”**

Conclusion

- Core Nodes (Planner, Radar, Vehicle Control) work well together. They are simple and very effective.
- Behaviour-Tree was the right choice for the problem at hand, but it created a lot of research- and implementation overhead. A more traditional solution (statecharts) might have yielded the same results in less time. (mostly because i have more experience designing those)
- Behavior Tree is a good choice as it helps debugging and understanding the system and gives structure
- Optimize global planner to use shortcuts in “no rules” mode
- Line-Detection works okay, but horizontal lines appear very late on the semantic segmentation camera. That makes stopping at a line hard. A HD-Maps-based approach would have been more stable.