Understanding the Autonomous System

Christopher Chong

Director of Product Manager, Automotive Microcontrollers & Processors

September 2018 | APF-AUT-T3319

SECURE CONNECTIONS
FOR A SMARTER WORLD

1.3 MILLION

Road traffic deaths occur every year

HIT BY A VEHICLE TRAVELING AT:	HIT BY A VEHICLE TRAVELING AT:	HIT BY A VEHICLE TRAVELING AT:
22 m/h	31 m/h	44 m/h
9 OUT OF **10** PEDESTRIANS SURVIVE*	**5** OUT OF **10** PEDESTRIANS SURVIVE	ONLY **1** OUT OF **10** PEDESTRIANS SURVIVES

OUT OF ALL ACCIDENTS GLOBALLY,

90% are caused by

HUMAN ERROR

Source: Seattle's Vision Zero Plan/Documents/Departments/beSuperSafe/VisionZeroPlan, ASIRT.ORG

Enabling Self-driving Cars

Better senses than the human driver.

Automation of driving decisions.

SENSE

THINK

ACT

Architecting the Car of the Future

More than a brain on four wheels.
The core of safe and secure mobility.

SENSE | THINK | ACT

LEVERAGING

Leadership in processing, security and mobile

NXP Core Values to Solve the Current Challenges of the Market

Computation Performance

Lead the heterogeneous compute performance with purpose built processors, optimized for power

Safety

No compromise on safety. Progression from ASIL to enhanced dependability and fail operational modes support

Ease of Use

Based on OPEN standards, portable and relocatable

Modularity Scalability

Built on ‘clear functionally separated extensible’ entities

Autonomous Driving

+$700 **Full** Automation — LEVEL 5

+$600 **High** Automation — LEVEL 4

+$400 **Conditional** Automation — LEVEL 3

+$100 **Partial** Automation — LEVEL 2

+$50 **Driver Assistance** — LEVEL 1

No Automation — 0

Level **3-5**
Automated Driving System
Performs entire dynamic driving task/monitors environment

Level **0-2**
Human Driver
Performs part of dynamic driving task/monitors environment

Semi Content per Car increase (TAM) vs Level 0
Source: Strategy Analytics; IHS; Evercore; ABI Research; NXP

Higher Levels of Automation: Higher Computation, Storage & Sensors

Typical Vehicle Architecture

Level	U-sonic	Radar	Imager	S.S. Lidar	V2X Cloud	Conv SIG. Proc/ML &/ DL	Storage
5 Level	8-12	6-12	8-12	1-3	Yes	AI >200k DMIPS, >50 TOPS	?
4 Level	8-12	6-12	8-12	1-3	Yes	AI 200k DMIPS, 50 TOPS	4TB
3 Level	8-12	4-8	3-8	0-1	Yes	AI 60k DMIPS, 5-10 TOPS	
2 Level	8-12	3-5	1-3	no		20k DMIPS, 1 TOPS	
1 Level	4-8	0-3	0-1	no		<10k DMIPS	

SAE J3016: driving automation levels

- DDT : Dynamic Driving Tasks
- ODD: Operational Design Domain
- Lateral, Longitudinal movement control
- OEDR: Object Event and Detection and Response
- DDT- FALLBACK
- MINIMAL RISK CONDITION
- REQUEST TO INTERVENE: Receptivity (of the driver)

Specified ODD

No ODD

DDT-Fallback is the DRIVER

DDT fallback-ready user must be immediately receptive

Request to intervene (xx Secs)

DDT-Fallback** is the **ADS

MAY Request to intervene

Partial OEDR
Driver MUST complement it

Complete OEDR
FULL perception and response

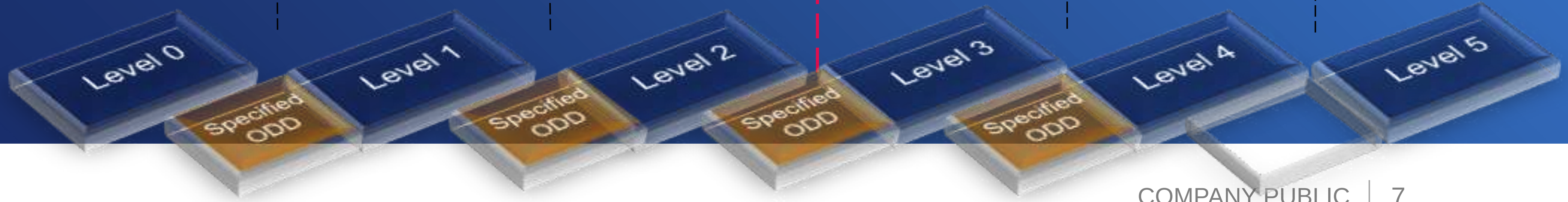
*Control of Lateral **OR** Longitudinal*

*Control of Lateral **AND** Longitudinal*

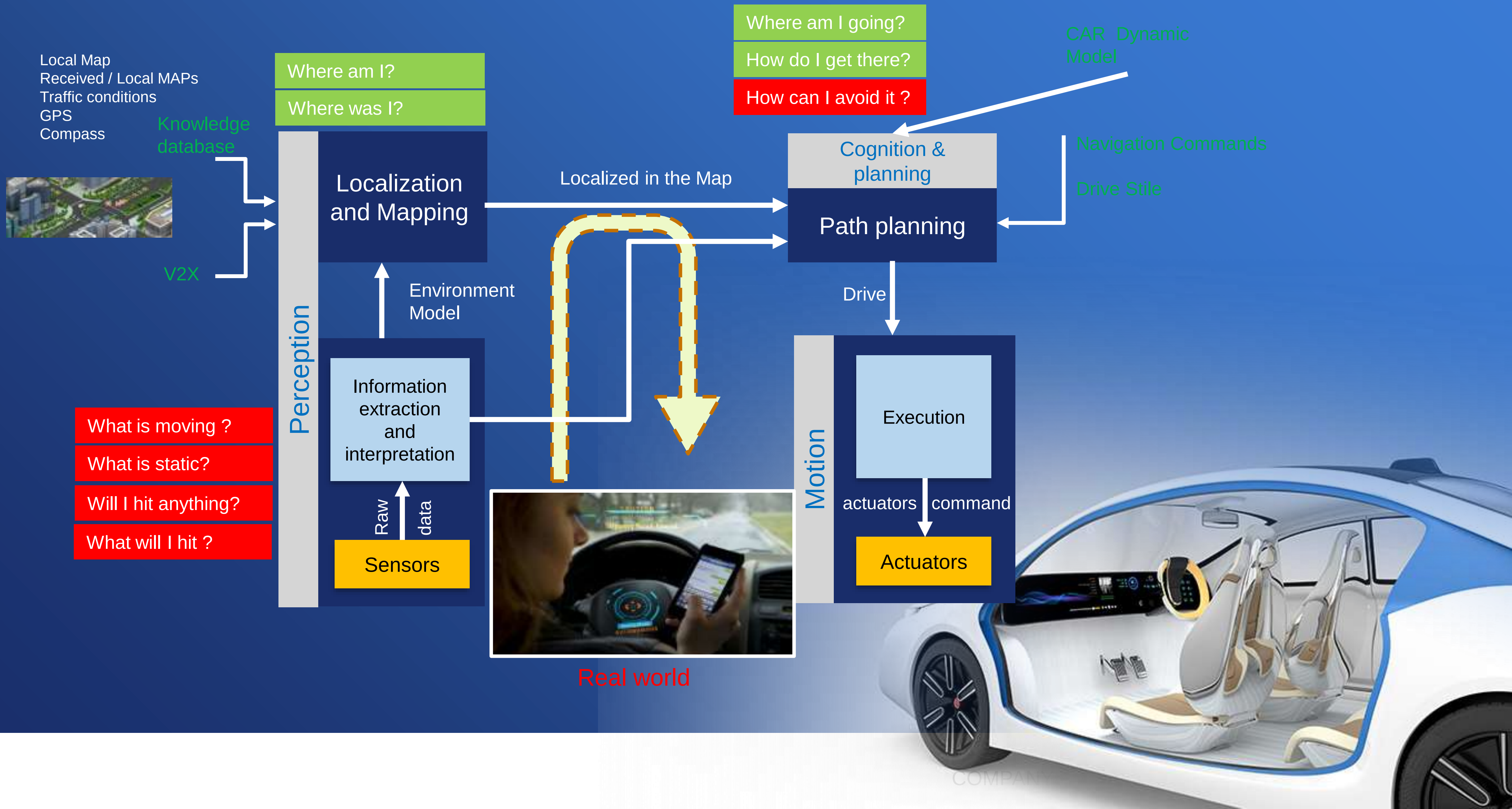
NO driving automation system

Sustained automation system

AUTOMATED DRIVING SYSTEM (ADS)



Autonomous Drive - Mobile Robot Systems



General Structure of ADS

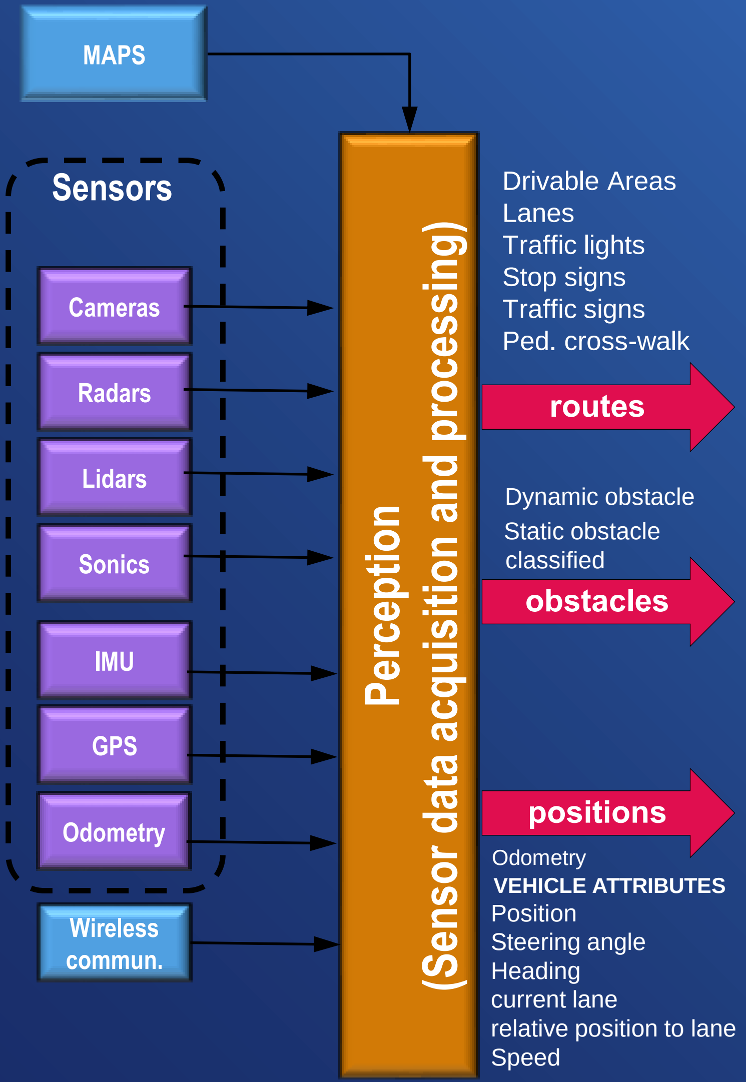
Perception	Localization and Mapping	Motion and Planning

Perception
- What is moving ?
- What is static?
- Will I hit anything?
- What will I hit ?

Localization and Mapping
- Where am I?
- Where was I?
- Where am I going?

Motion and Planning
- How do I get there?
- How can I avoid it ?

Environment Perception scope

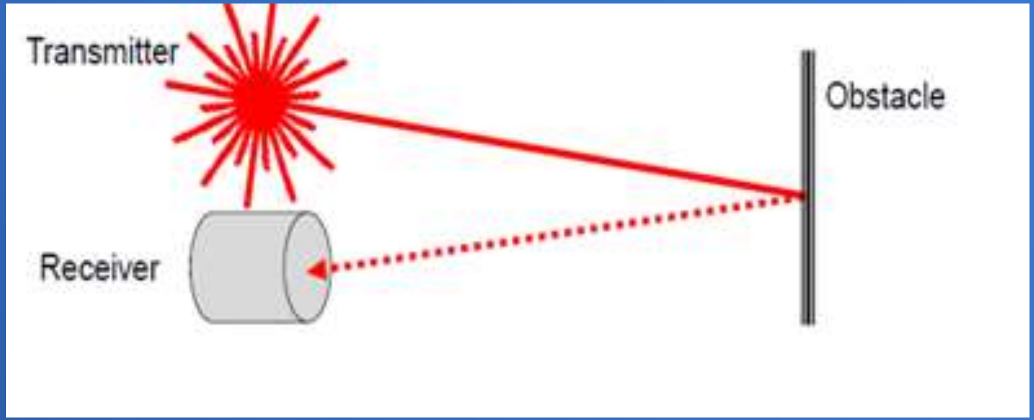


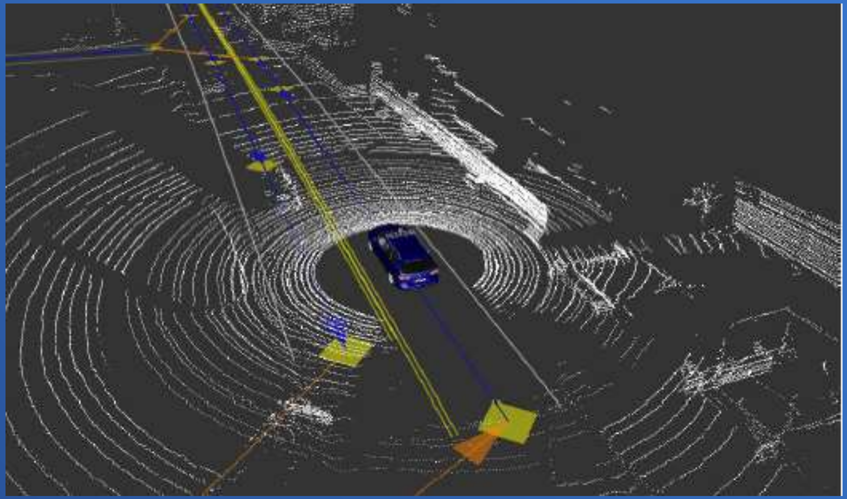
- **Static layer**: 3D static objects (from sensors) and road elements (from sensors and Map)
- **Dynamic layer**: Independently moving objects (cars, pedestrians, bicycles…).
- **Localization layer**: Accurate vehicle localization (pose)
  - **MAP-driven approaches**: Control the vehicle's pose in relation to a global coordinate system.
    - The map data is used to provide information about the stationary environment, especially about the course of the lanes.
    - Even more detailed maps are used to improve the vehicle's global map-relative position
  - **Perception-driven approaches**: Perceive the complete environment with on-board sensors.
    - Sensors data are fused together to determine accurate distances and speed of relevant objects.

Perception - LIDAR

LIDAR

- Range up to 100-200 meters
- Field of View FOV up to 360 deg Horizontal
  0-30 deg Vertical
- Accuracy: 5 to 1.5 cm, 3 - 0.1 degree

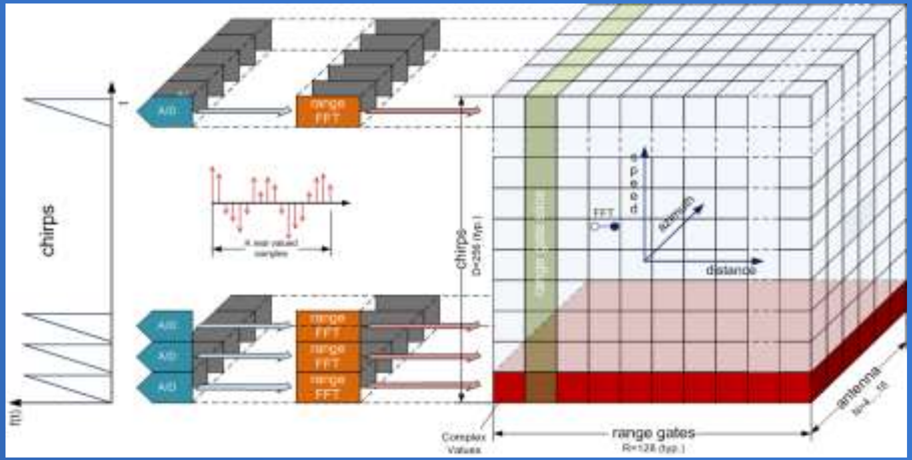


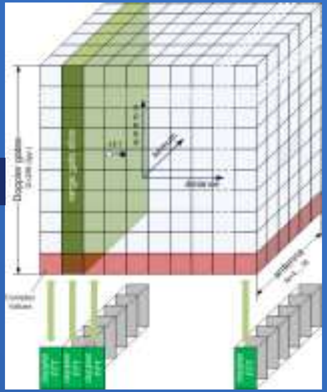


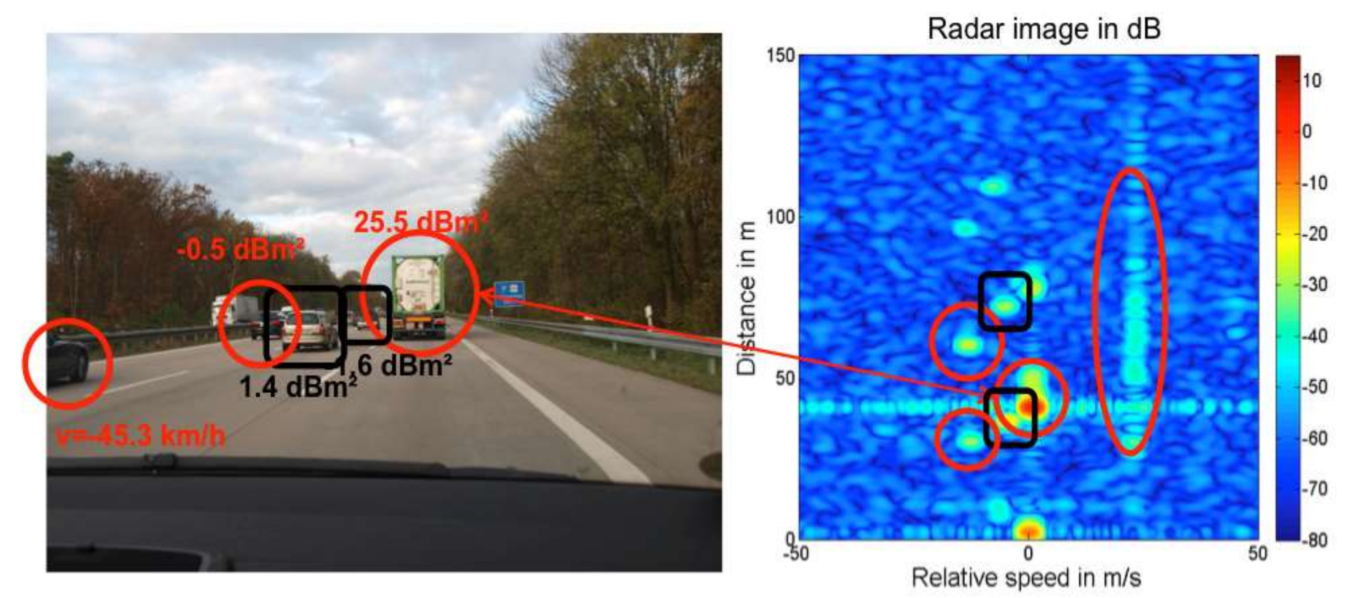
Perception - Radar

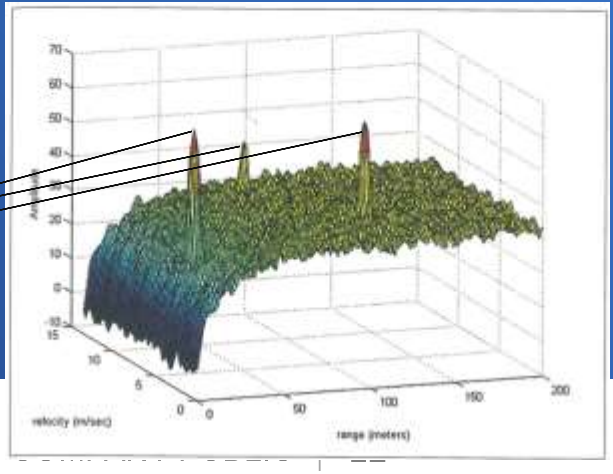
RADAR

- short range (30 - 70 meters)
- long range (70-200+ meters)
- FOV V:5degree H: 60 degree
- accuracies 5cm, 0.5m/Sec (~18 Km/hr), 1-3 deg





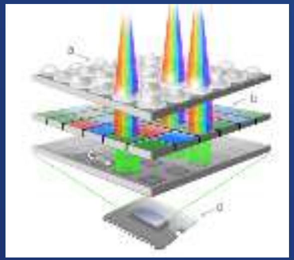


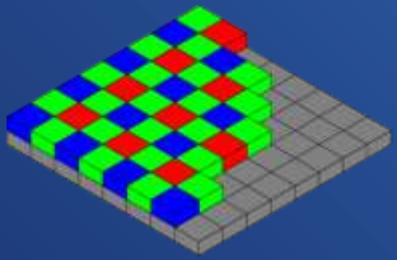


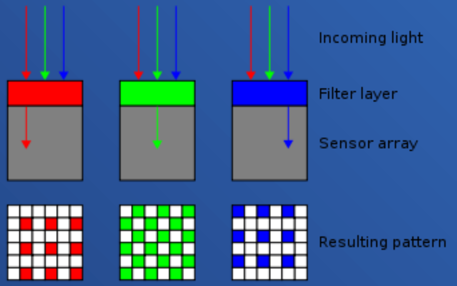
targets

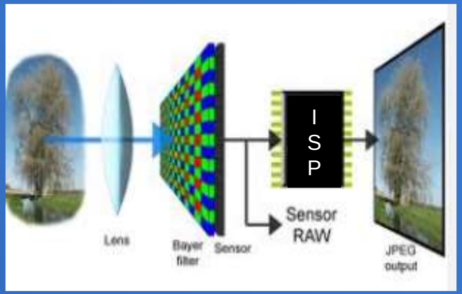
Perception - Camera

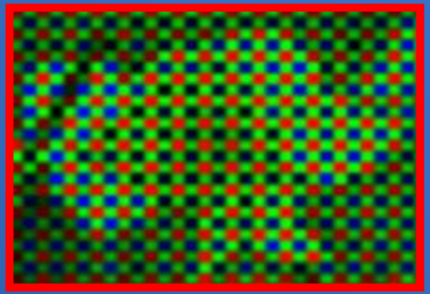
CAMERA
- 30/60/120 Megabyte/sec
- Range: 3 to 100 meters
- Field of View strongly depend on range
- Resolution depends on sensor/optics/distance
  (pix 3.75um, f: 5.7mm, d:70m ~50 mm)











Sensors for Perception

NATURAL DISTANCE MEASUREMENT
Short-term planning collision avoidance

LIDAR

- 600K 3D/2D points/sec (2.5 Gbyte/sec)
- Range up to 100-200 meters
- Field of View FOV up to 360 deg Horizontal
  0-30 deg Vertical
- Accuracy: 5 to 1.5 cm, 3 - 0.1 degree

Advantages:

- 3-D map of the world around you. Can generate ROAD MAP
- Works independent of the ambient light
- It is robust against interference

Drawbacks:

- Expensive (today)
- In hot conditions, Lidar doesn't distinguish between a big dust cloud and a brick wall
- Limited information about texture or color of surfaces

RADAR

- 500 Kbyte/sec
- short range (30 - 70 meters)
- long range (70-200 meters)
- FOV V:5degree H: 60 degree
- accuracies 5cm, 0.5m/Sec (~18 Km/hr), 1-3 deg

Advantages:

- Is virtually immune to problematic visibility and lighting conditions
- Has a good accuracy in longitudinal distance measurement
- return speed and distance

Drawbacks:

- Challenged for identifying non metallic objects.
- No information about texture or color (no TL, no TSR)
- Has poor accuracy in lateral measurement

ULTRASOUND

- Few Kbyte/sec
- VERY short range (1 - 3 meters)

Advantages:

- VERY CHEAP

Drawbacks:

- Low range
- Low speed
- Low resolution

Requires
INTERPRETATION
Mid-Term hazard
avoidance

CAMERA

- 30/60/120 Megabyte/sec
- Range: 3 to 100 meters
- Field of View +/- 50
- Resolution depends on sensor/optics/distance
  (pix 3.75um, f: 5.7mm, d:70m ~50 mm)

Advantages:

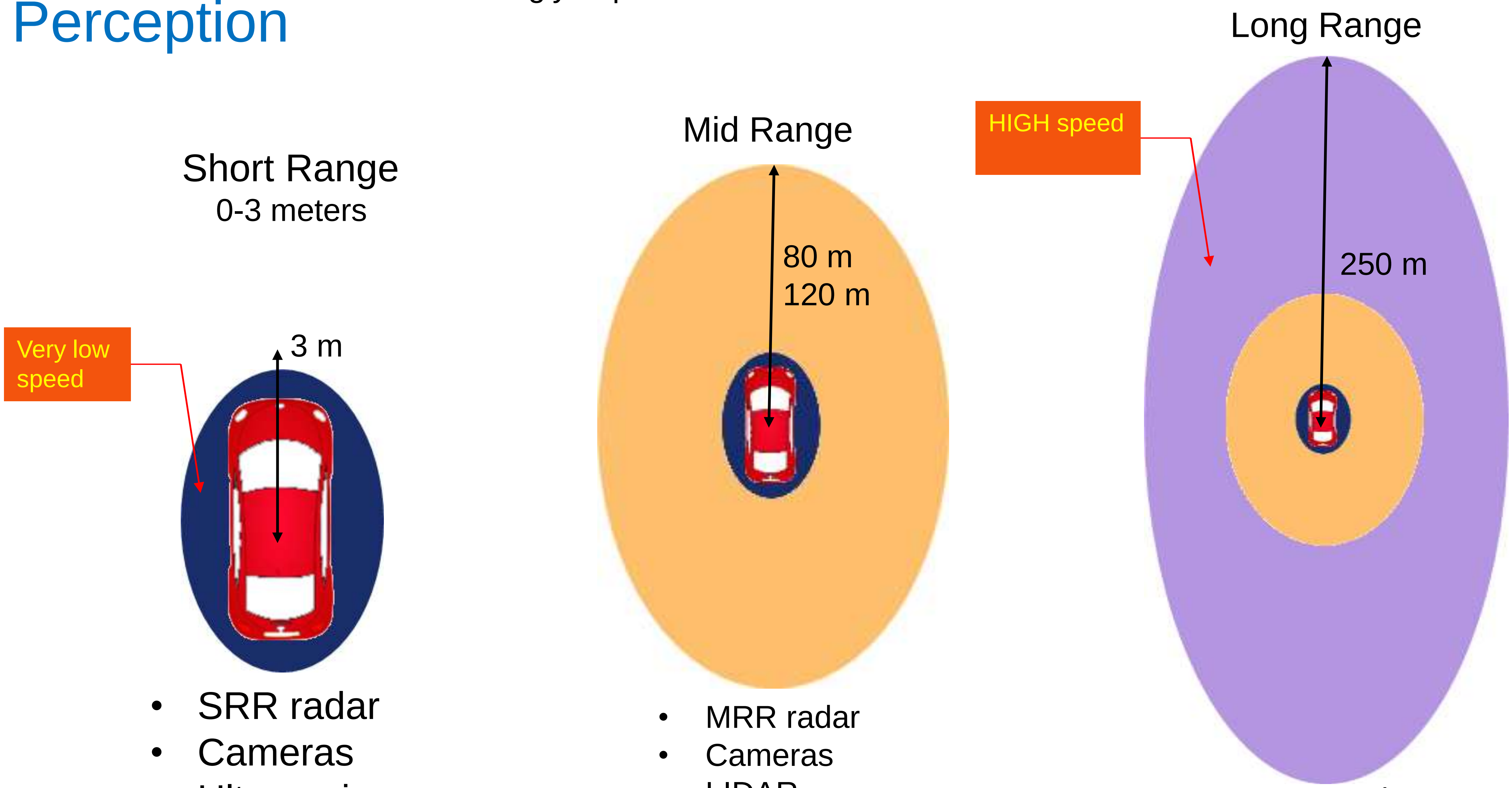
- Inexpensive
- Works by received lights (recognize color and pattern)
- Has good accuracy in lateral measurement
- RICH informations
- Semantic Understanding

Drawbacks:

- Challenged by lights conditions
- Challenged by weather conditions
- Need Multiple-camera systems to provides reliable 3D information
- Difficult interpretation

Perception

strongly depends on the ADS ODD and level.

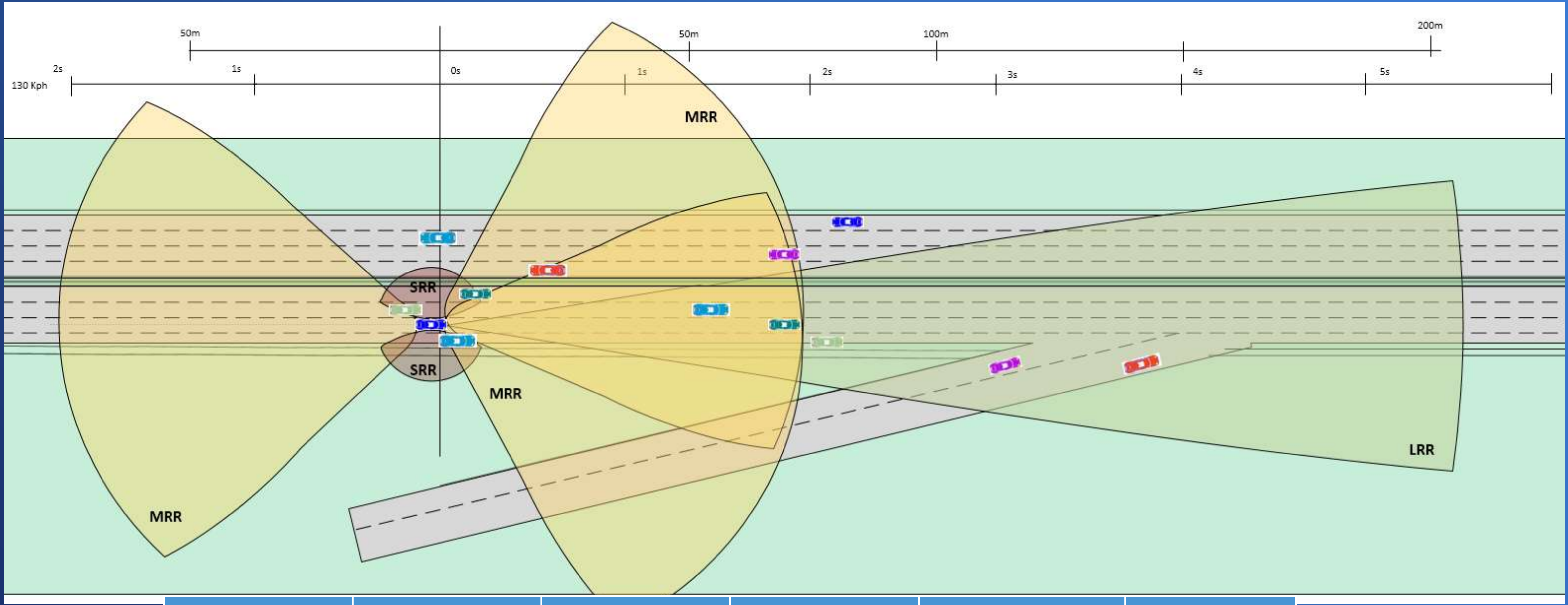


- SRR radar
- Cameras
- Ultrasonic

- MRR radar
- Cameras
- LIDAR

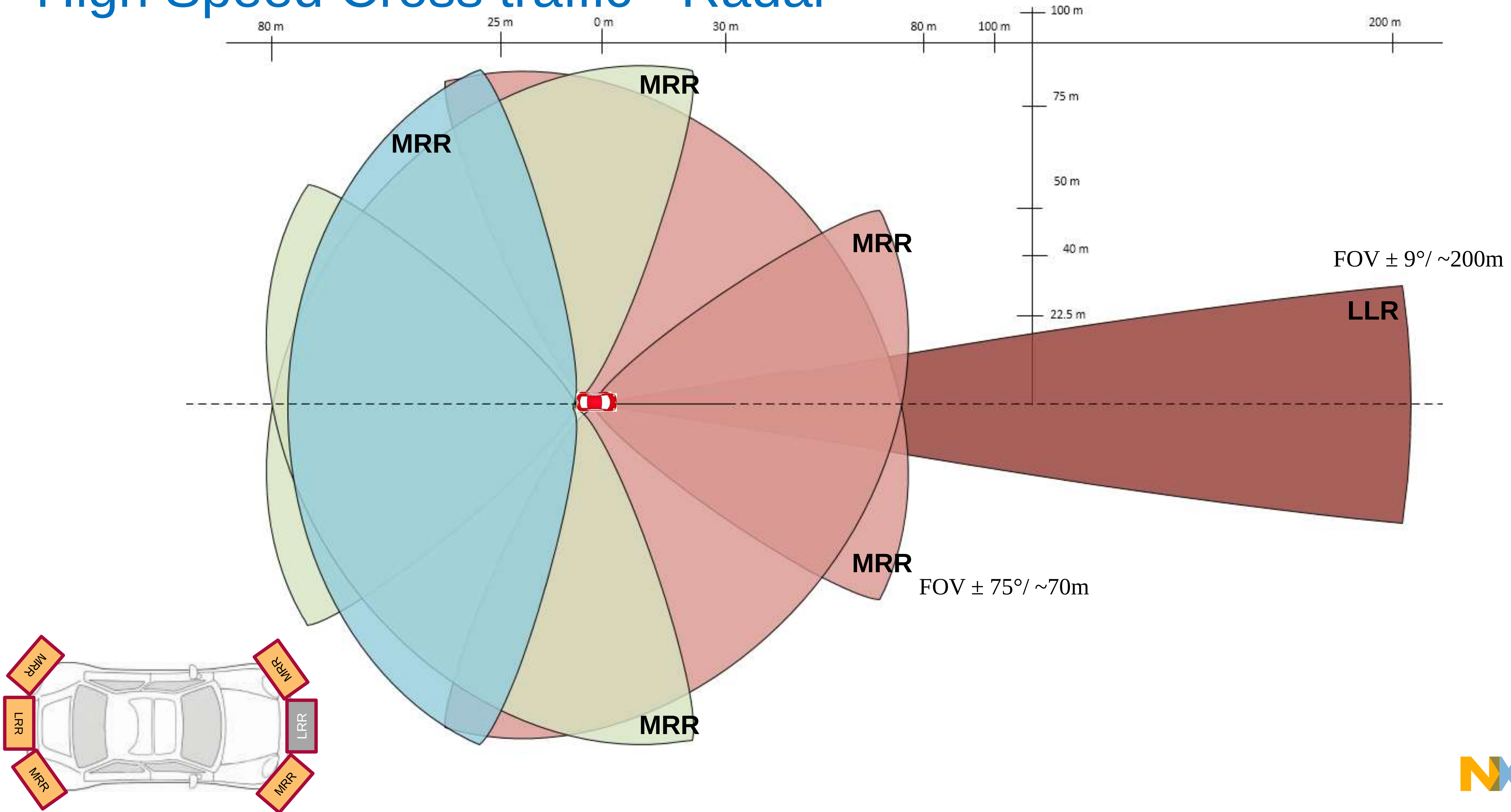
- LRR radar

Highway Pilot – radar based

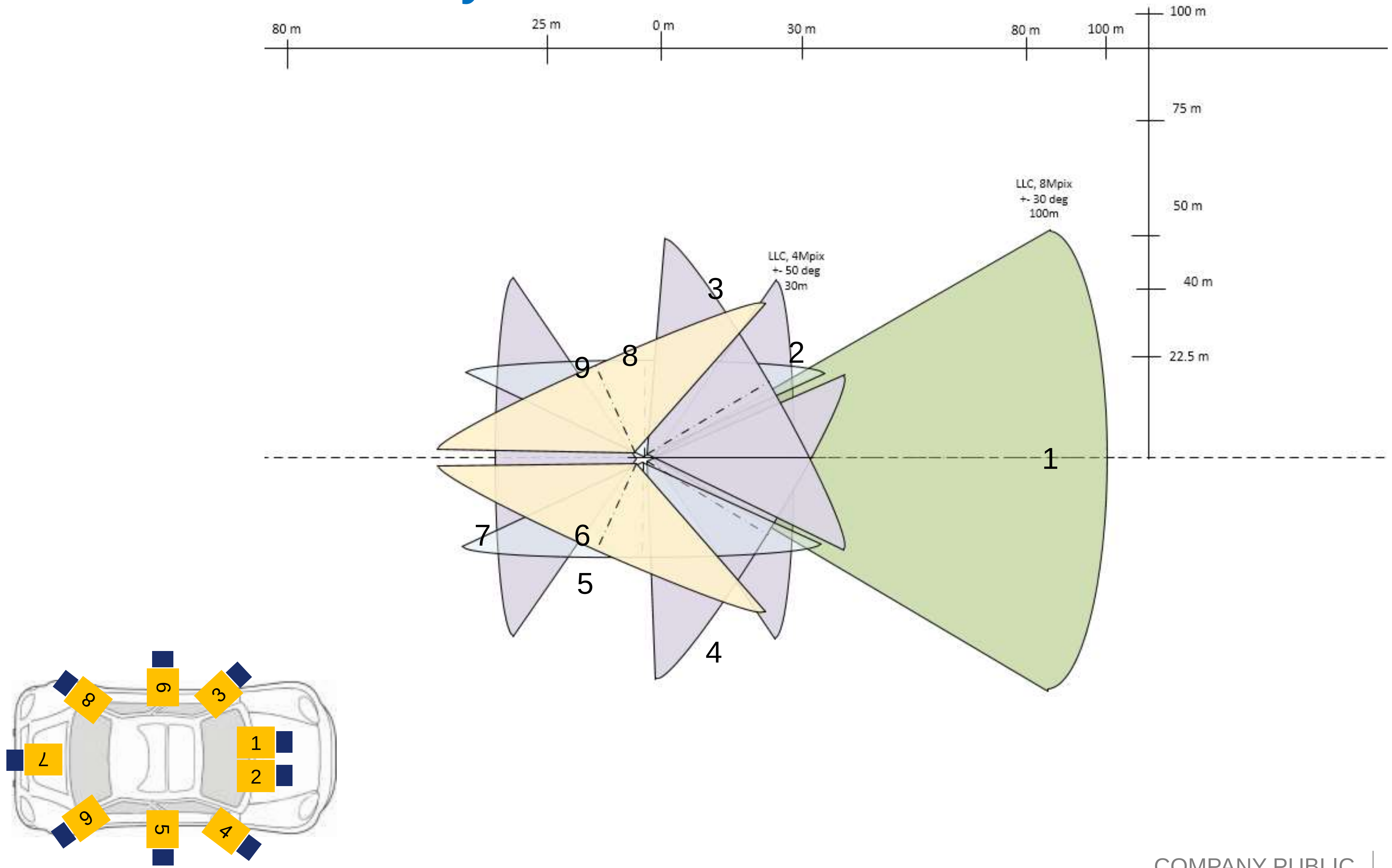


ADS reaction	Max Speed	Min stop distance btw vehicle	Emergency break time	Emergency Stop time	Time Horizon
150 mSec	130 km/h 36 m/s	6m	4.2s	~80m	200m: 5.5s
150 mSec	260 km/h	11m	8.3s	~231m	300m: 4s

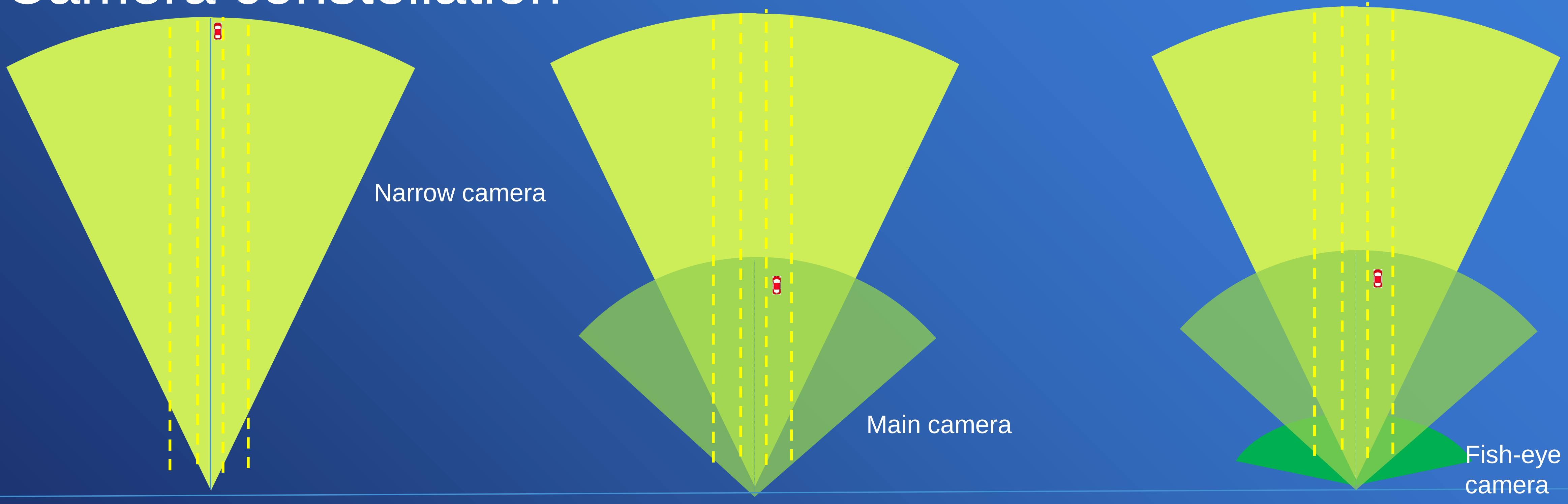
High Speed Cross traffic - Radar



Sensor Diversity - Cameras



Camera constellation



FOV ± 12.5°/ ~ 135 m (8MP)
/ ~ 96 m (4MP)
/ ~ 70 m (2MP)

- Objects (car, pedestrian, bicycles)
- Lanes
- Traffic Lights
- Debris
- Disparity with Main to get redundancy

FOV ± 25° / ~65 m (8MP)
/ ~48 m (4MP)
/ ~35 m (2MP)

- Objects (car, pedestrian, bicycles)
- Lanes
- TSR, AHB, Traffic Lights
- Path delimiter
- Used for Lateral control assist

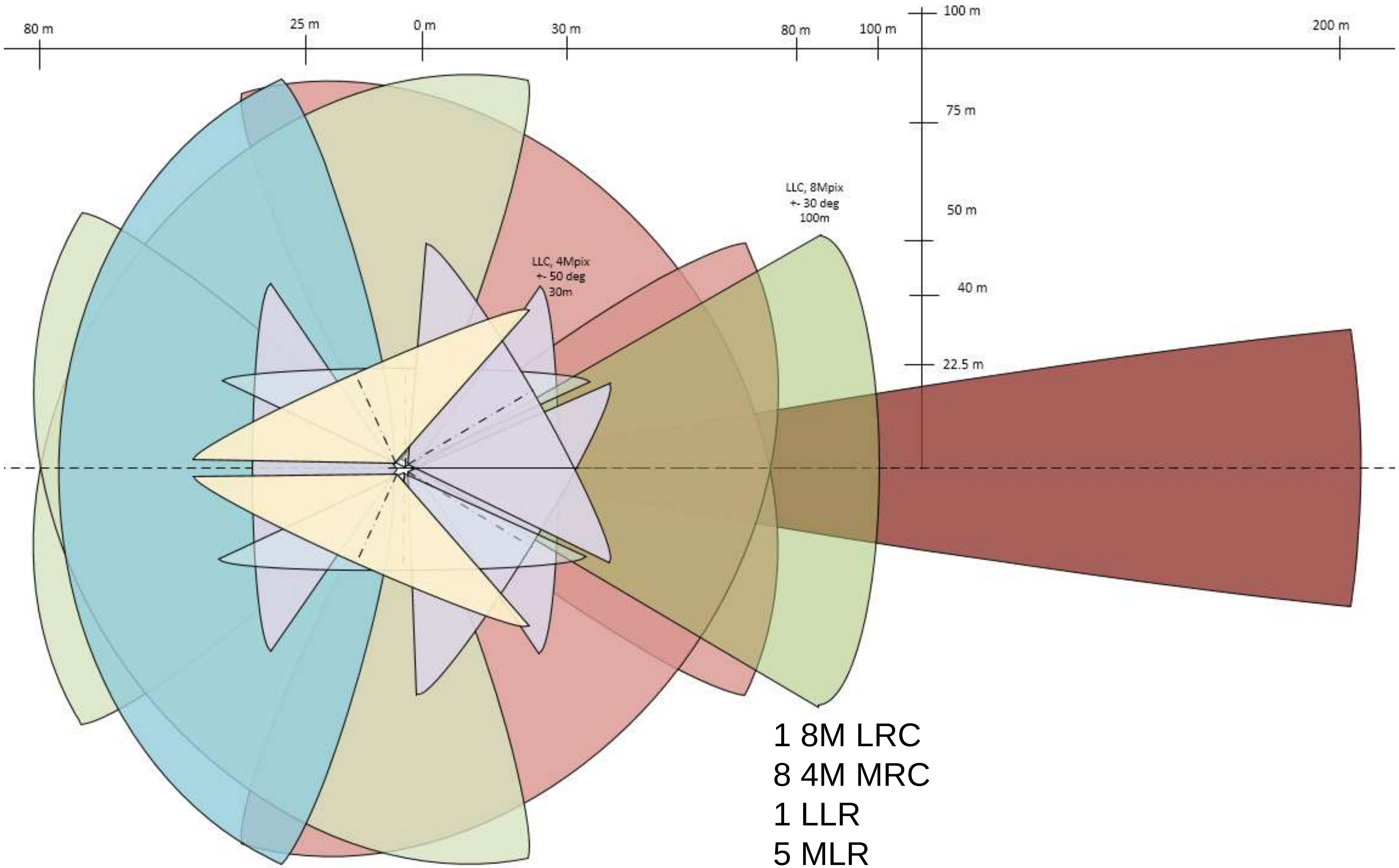
FOV ± 70° / ~11 m (8MP)
/ ~ 8 m (4MP)
/ ~ 6 m (2MP)

- CLOSE Objects (car, pedestrian, bicycles)
- Lanes for tights curves
- TSR, AHB, Traffic Lights
- Cut in detection
- First in row Traffic Light

High-speed Multisensor svstem

- Based on ODD and Lev: * operational coverage, redundancy and diversity

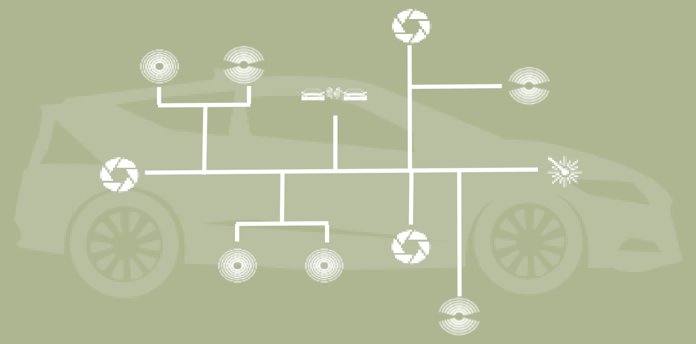
- Based on Level, how cope with sensor failure



1 8M LRC
8 4M MRC
1 LLR
5 MLR

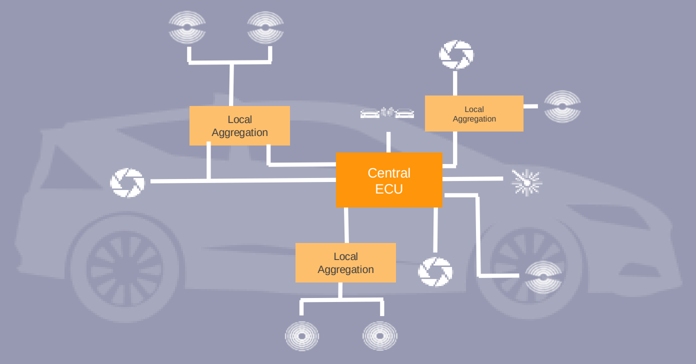
Architectures Topologies

Distributed
No central fusion unit



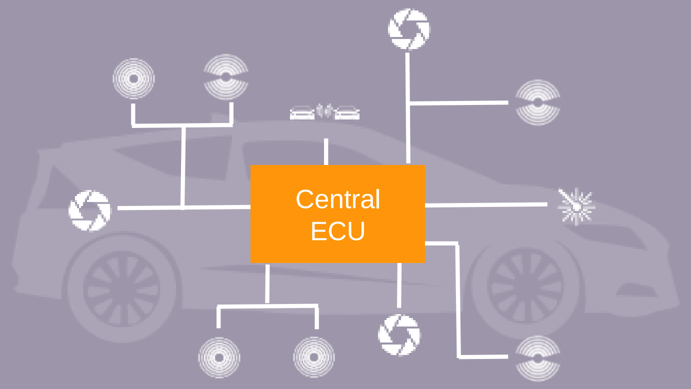
Not likely: redundant ECUs
- Each ECU does fusion at object level
- Each ECU knows all others data
- Lower BW for 'fused' data to be exchanged
- Simultaneity required
- Increase sensors costs and complexity

Hybrid
Central fusion & smart sensors



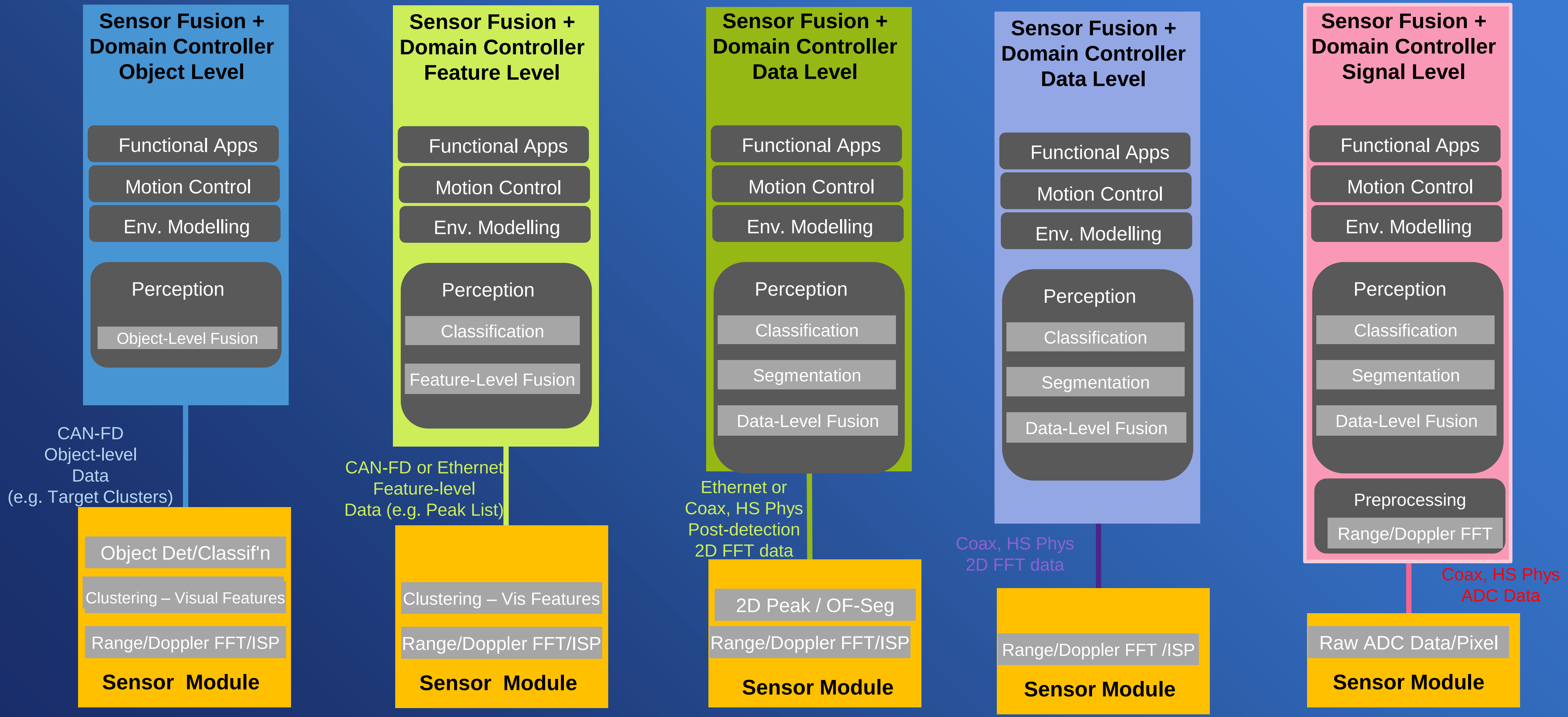
Most likely: flexible and Scalable
- Leverage Smart Sensing and local Aggregation with a scalable Central Fusion
- Manages costs of Data distribution vs. precision for fusion
- Distributes effort and eases interoperability

Central Super computer
Central server and naïve sensors

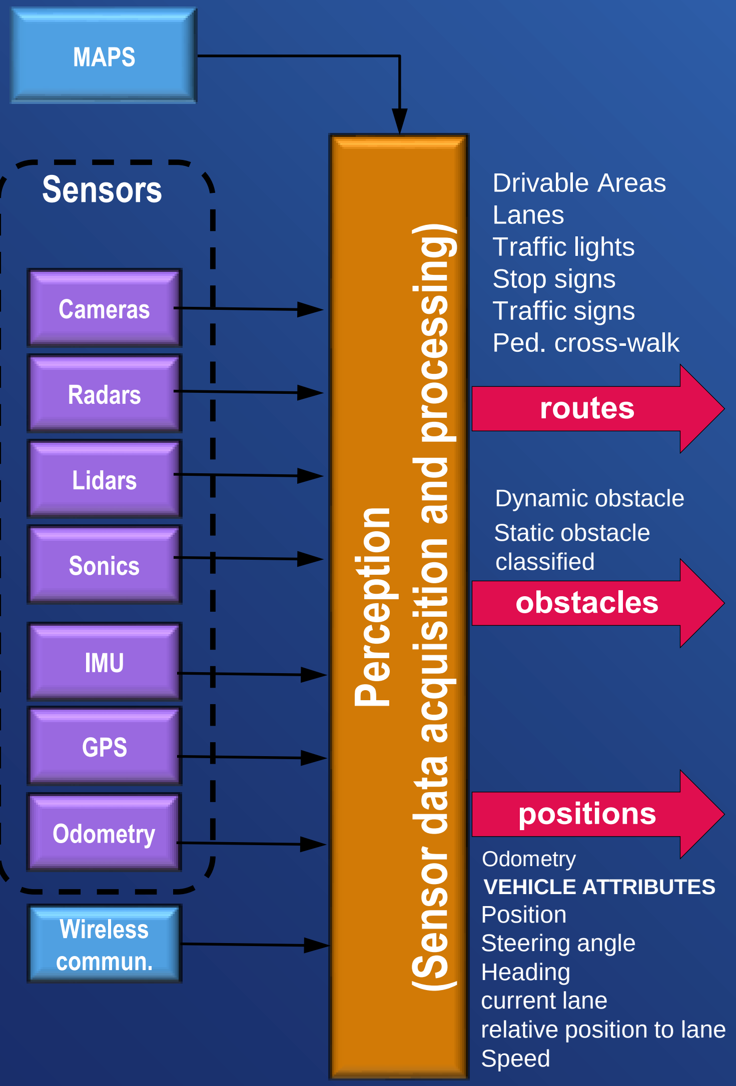


Likely: complexity vs. cost
- Requires industry level interoperability
- Early Fusion highly wished but little done to date due to complexity
- Best Sensor Data fusion in theory
- Extra large BW for raw data to be TX-ed
- Sensors are simplified and cheaper potentially (but Sensors vendors fight against this trend)

Sensor Sub-System Architectures



Environment Perception scope



- **Static layer**: 3D static objects (from sensors) and road elements (from sensors and Map)
- **Dynamic layer**: Independently moving objects (cars, pedestrians, bicycles…).
- **Localization layer**: Accurate vehicle localization (pose)
  - **MAP-driven approaches**: Control the vehicle's pose in relation to a global coordinate system.
    - The map data is used to provide information about the stationary environment, especially about the course of the lanes.
    - Even more detailed maps are used to improve the vehicle's global map-relative position
  - **Perception-driven approaches**: Perceive the complete environment with on-board sensors.
    - Sensors data are fused together to determine accurate distances and speed of relevant objects.

Localization example

GPS/IMU can be used

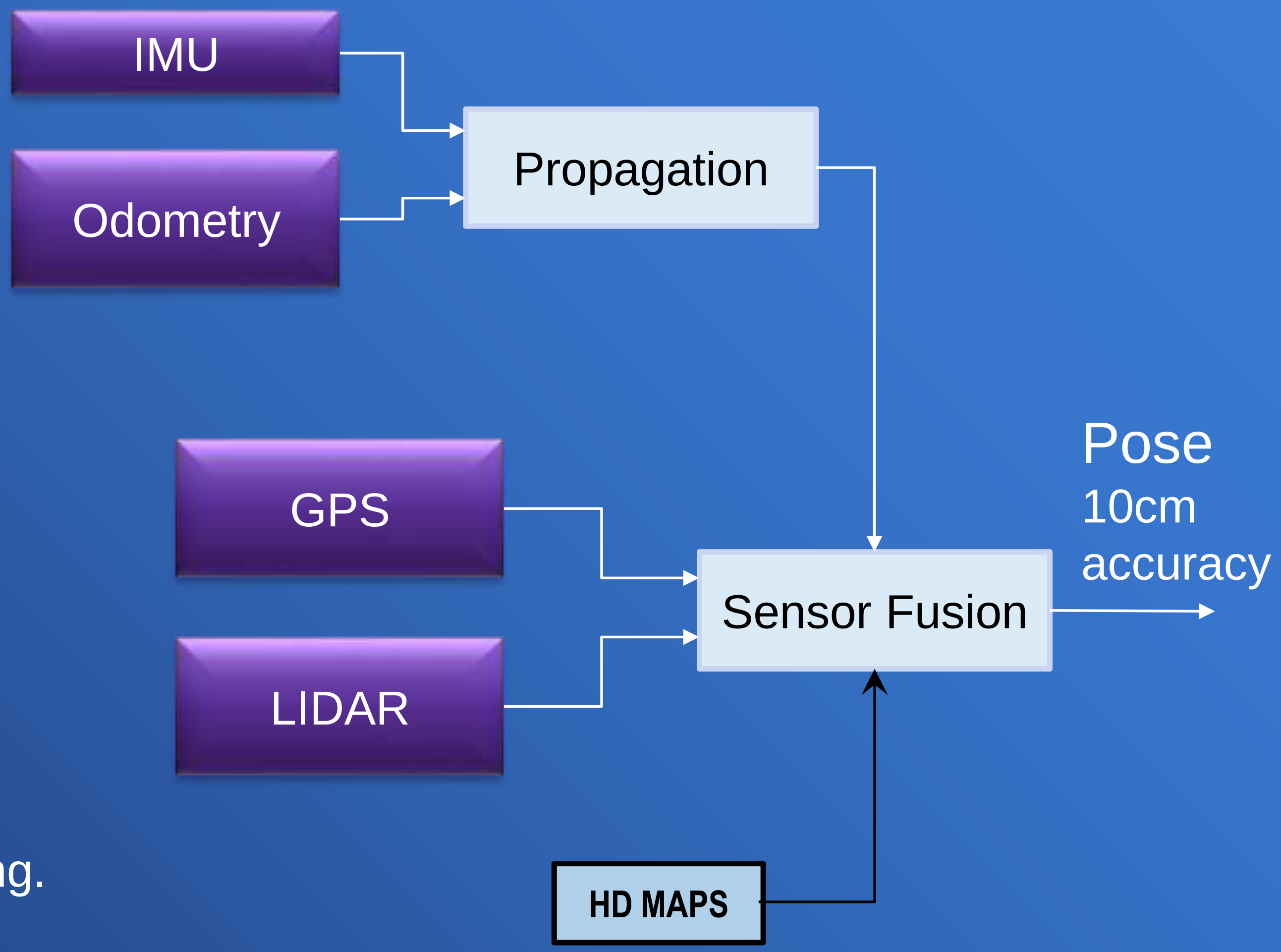
- GPS accurate at 1-5 m and slow update
- IMU inaccurate and fast update
- Kalman filter to combine the two

But unfortunately

- Accuracy 1 m achieved is not enough
- GPS noise sensible to buildings
- Not working in tunnel

Cameras can be used for localization as well considering an EGO Motion estimation (via stereo cameras or Mono Optical flow) and feature matching. Approach very sensitive to light.

Lidar can also be used and compared with HDMAP using Particle Filters

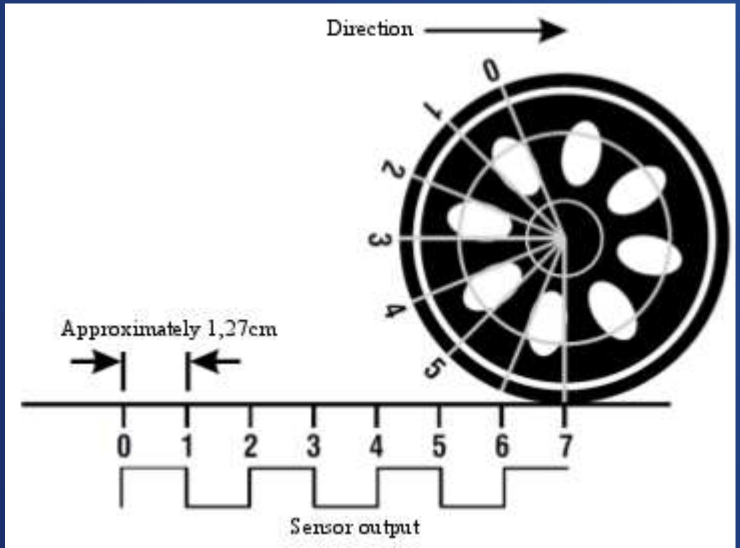


Wheel Odometry

Dead Reckoning is the process of calculating vehicle current position using a previously determined position through known course and velocity information over given length of time.

The most simplistic way is **Wheel encoders** to measures wheel rotation and/or steering orientation. The idea is to integrate over time that also cumulate the errors.

Fusing them with other sensors we can improve the accuracy.



Wheel odometry suppose rotation is translated to linear displacement. That suppose DRIFT will introduced further errors:

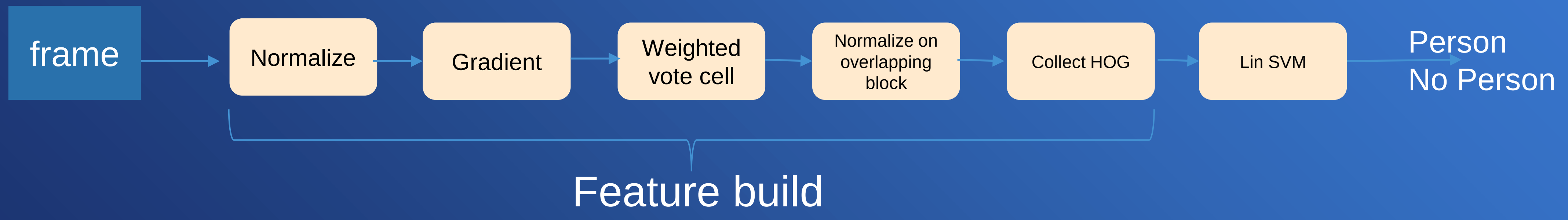
- Systematic errors like change in wheel diameter. It accumulates constantly.
- Non-systematic errors: uneven floor, wheel slippage due to oil or acceleration. May appear suddenly and add huge errors.

Detection and Classification

Autonomous vehicle share road with many other traffic participants.

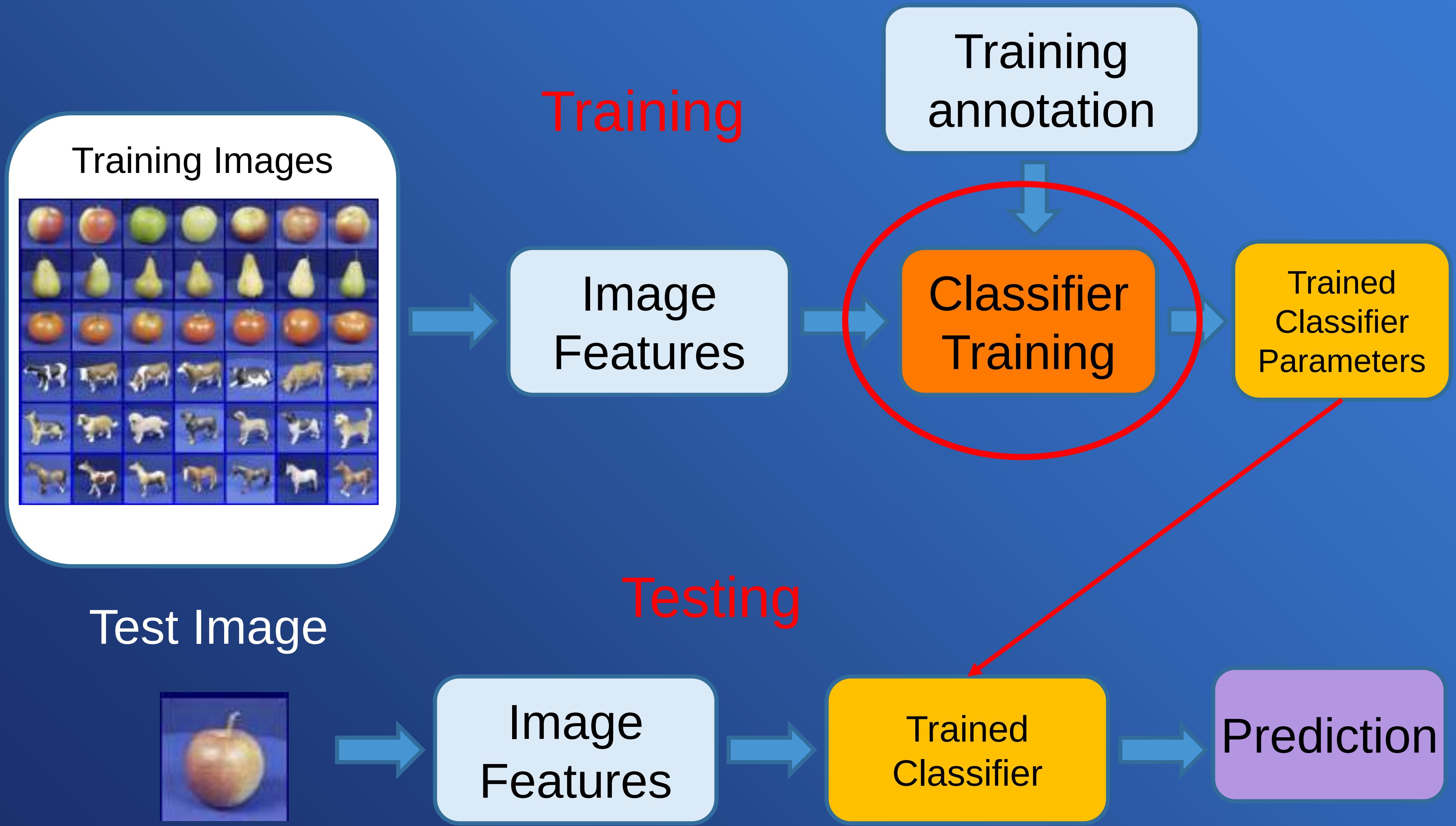
There are also obstacles, lane dividers, and many other objects on the road

Fast and reliable detection is crucial for safety reasons.



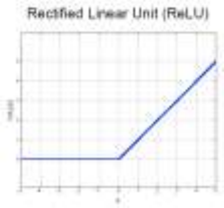
Detection and classification can also be based on Lidar and Radar but mostly rely on Vision.

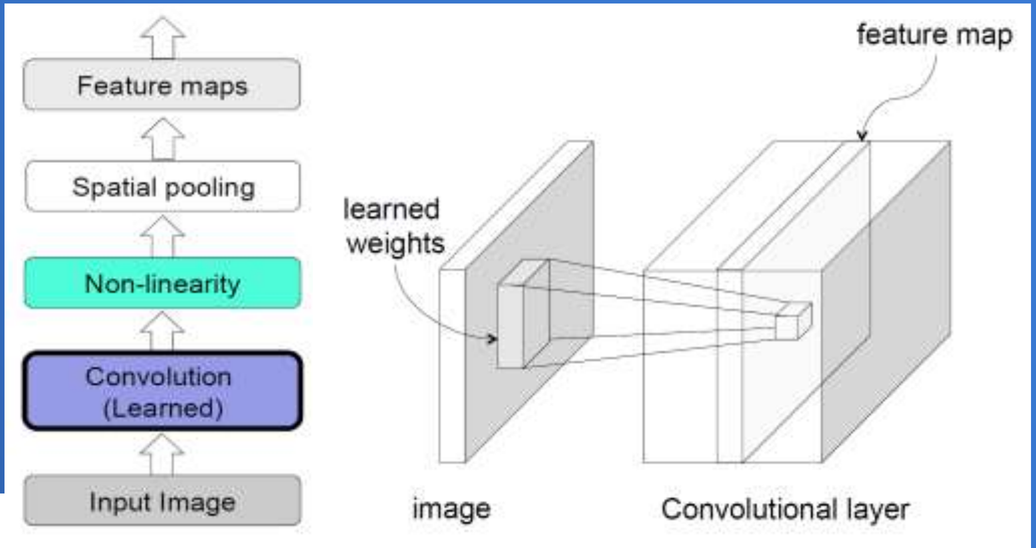
Machine Learning and Classification



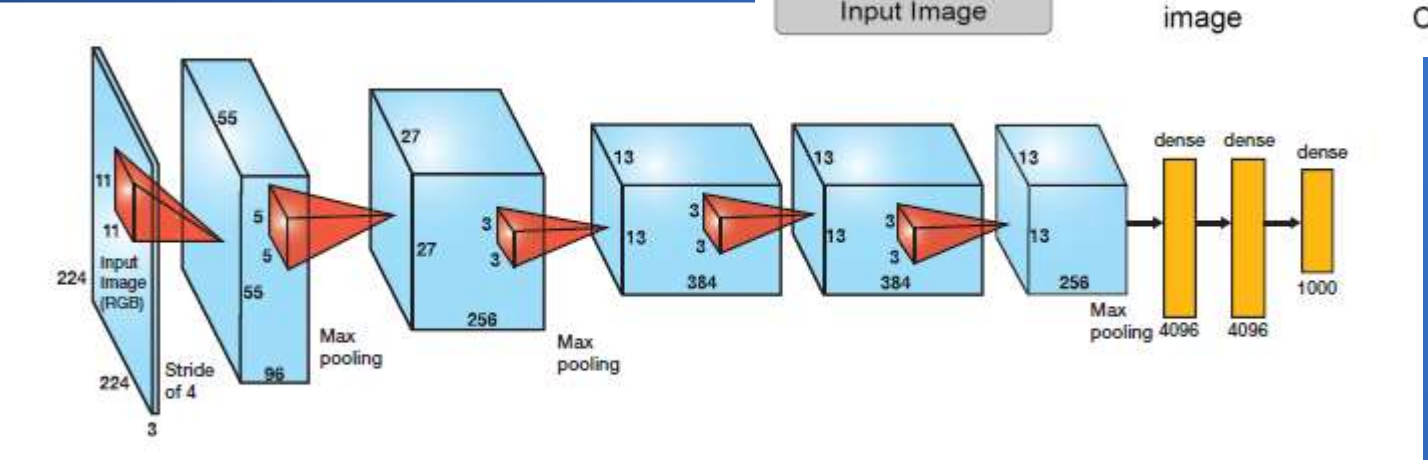
Conv. Neural Network

Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity.



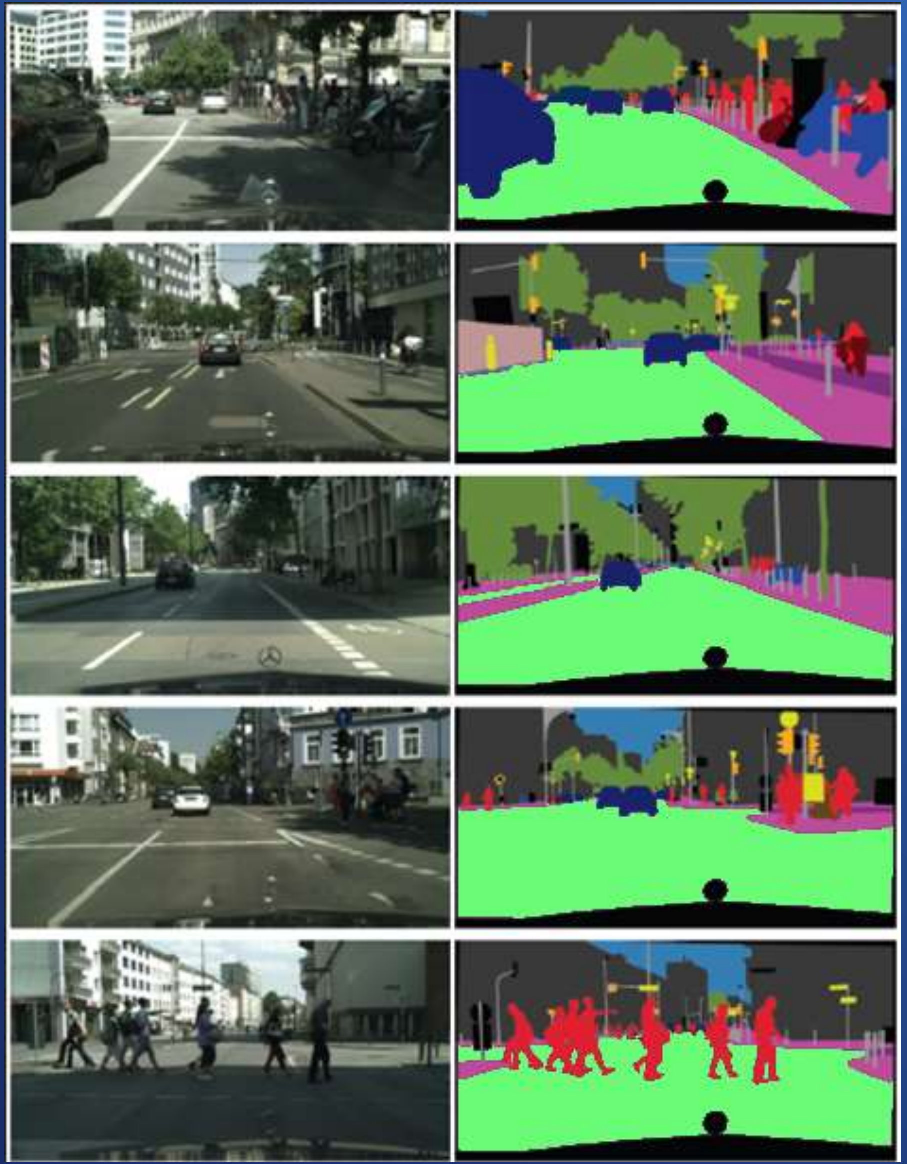


AlexNet (2012)



1. Region Proposal (R-CNN); e.g. using 3x3 sliding windows, different scales (1:1, 1:2,2:1) of 3x3=9 combination are considered. On a 1000 x 600 this is 20000 hypothesis. CNN will make this very efficient and reduce redundancy (max suppression) to ~2000 proposals.
2. The proposed window is projected into a fix-size feature map by a ROI pooling layer

Semantic Segmentation



Parsing image from camera into semantic meaningful segment gives autonomous vehicle structured understanding of its environment

Semantic segmentation predicts classes labeling dense part of the image.

Environment Perception scope

MAPS

Sensors
- Cameras
- Radars
- Lidars
- Sonics
- IMU
- GPS
- Odometry

Wireless commun.

Perception
(Sensor data acquisition and processing)

Drivable Areas
Lanes
Traffic lights
Stop signs
Traffic signs
Ped. cross-walk

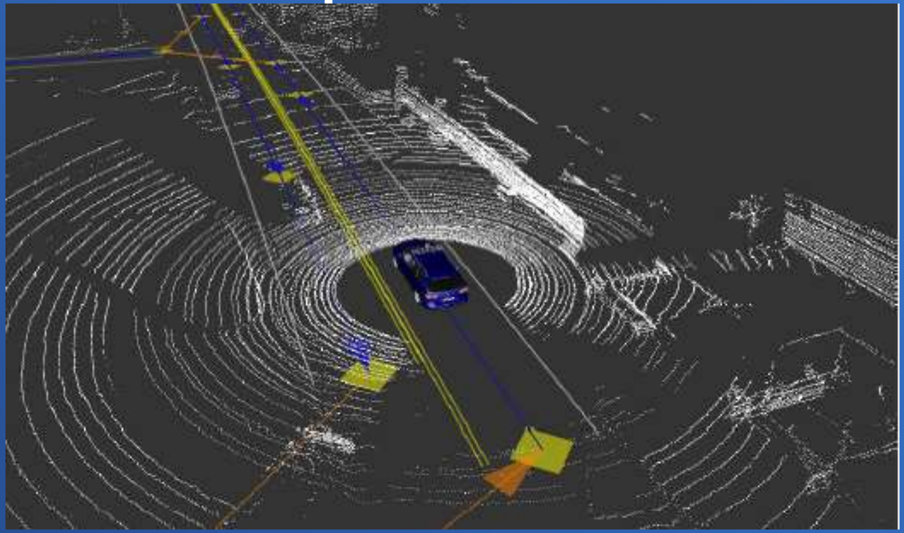
routes

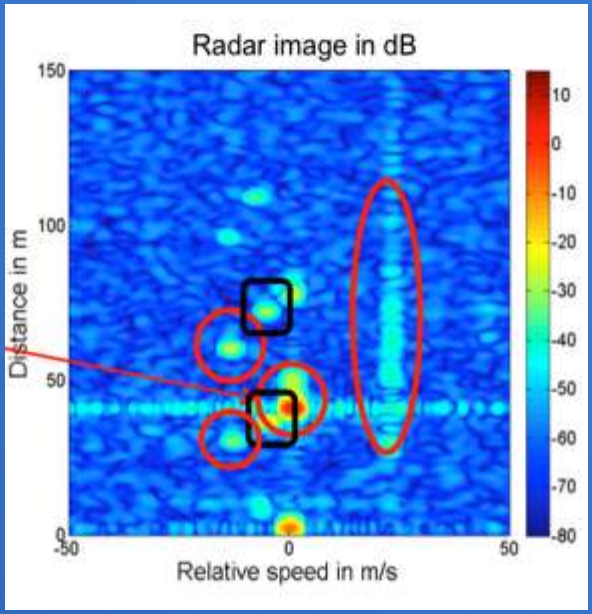
Dynamic obstacle
Static obstacle
classified

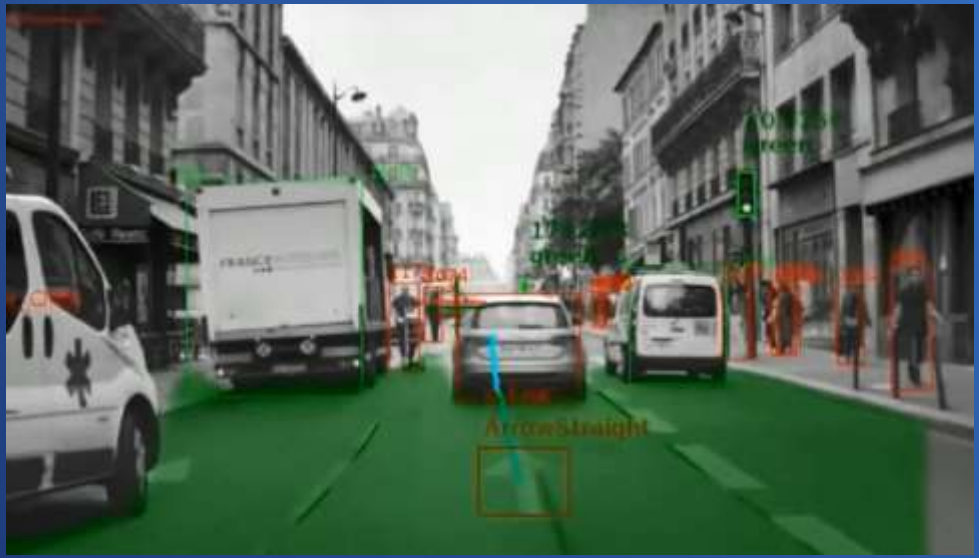
obstacles

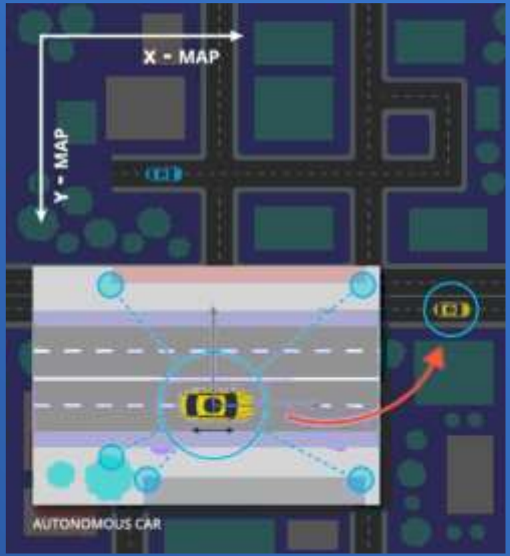
positions

Odometry
VEHICLE ATTRIBUTES
Position
Steering angle
Heading
current lane
relative position to lane
Speed









Traffic Prediction

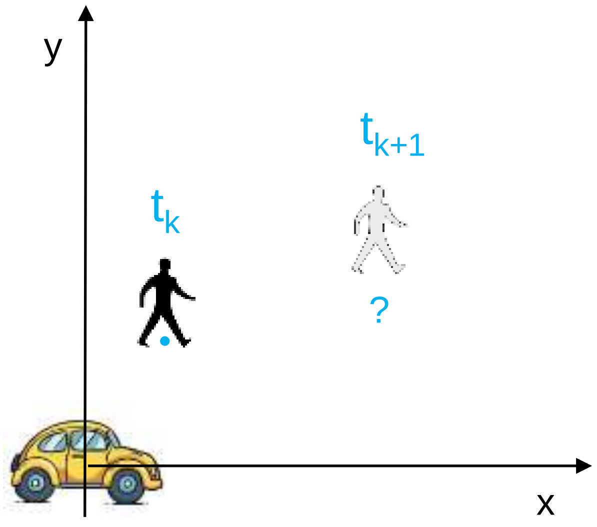
Predict the behavior of the detected perception objects in the near future.
Output spatial-temporary trajectory points.

Traffic Prediction is trying provide for the detected dynamic objects 2 attributes:

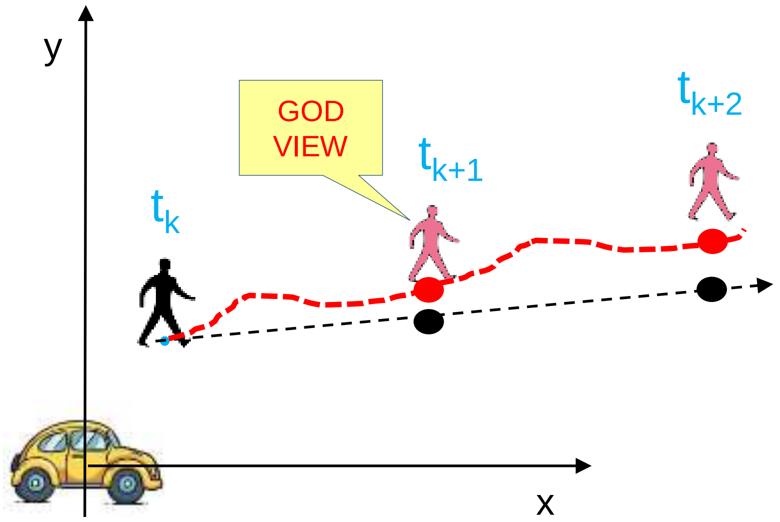
- **Kinematic**: Considering simple physical models (different for different objects) an immediate prediction related with attributes like position, velocity, heading, acceleration…. can be done. Result is in practice a predicted trajectory for each detected moving object. This module can be implemented on the perception side or in the planner side, depending on the nature of the sensors and the fusion mechanism.

- **Behavior**: the objective of T.P. is not just immediate prediction given the physical attributes but more about a behavioral level that span for a periods of few seconds (5+ secs). This is where machine learning based approach are mostly used.

Traffic Prediction - Kinematic



Traffic Prediction - Kinematic



State pedestrian

$$\boldsymbol{x} = \begin{pmatrix} px \\ py \\ vx \\ vy \end{pmatrix}$$

Formal physical description of the object attributes

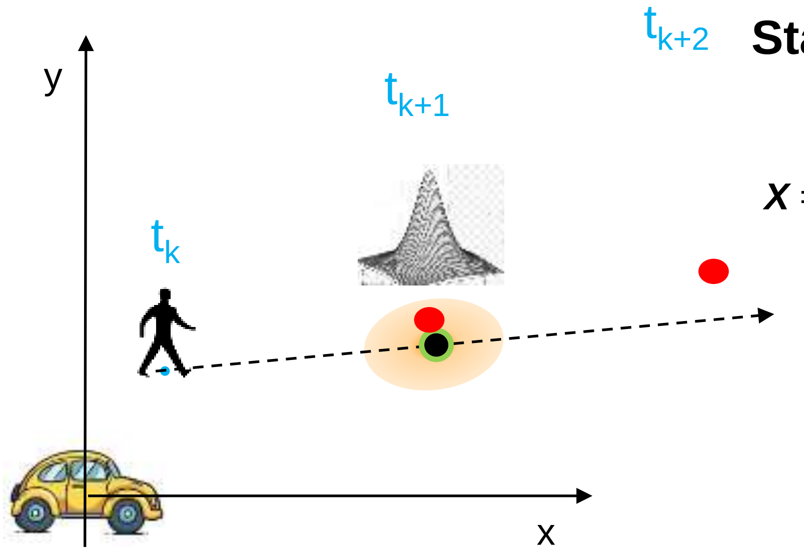
Model of pedestrian motion

$$x_{k+1} = f(x_k)$$

$$\begin{cases} px_{k+1} = px_k + vx_k * \Delta t \\ py_{k+1} = py_k + vy_k * \Delta t \\ vx_{k+1} = vx_k \\ vy_{k+1} = vy_k \end{cases}$$

Model is a physical approximation of real world

Traffic Prediction - Kinematic



State pedestrian

$$\boldsymbol{x} = \begin{pmatrix} px \\ py \\ vx \\ vy \end{pmatrix}$$

Formal physical description of the object attributes

Model of pedestrian motion

deterministic

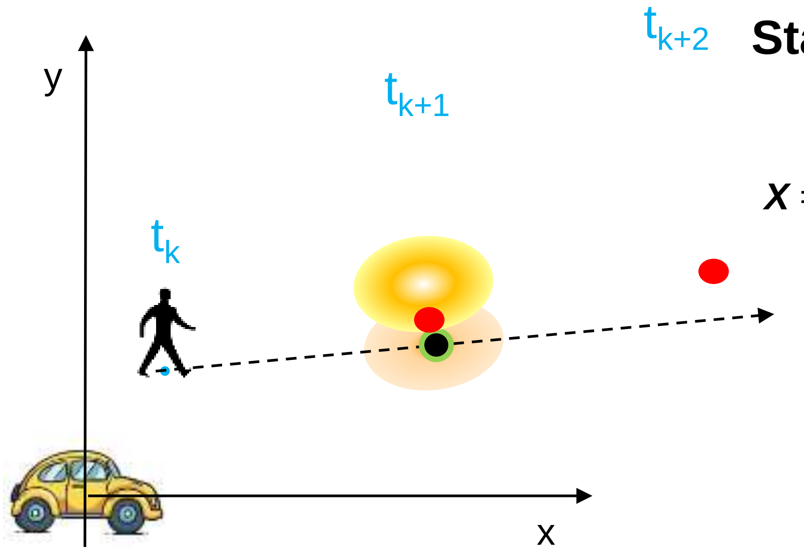
Stochastic

$$\boldsymbol{x_{k+1}} = \boxed{\boldsymbol{f}(\boldsymbol{x_k})} + \boxed{\boldsymbol{N}}$$

$$\begin{cases} px_{k+1} = px_k + vx_k * \Delta t + Nx \\ py_{k+1} = py_k + vy_k * \Delta t + Ny \\ vx_{k+1} = vx_k + Nvx \\ vy_{k+1} = vy_k + Nvy \end{cases}$$

Model is a physical approximation of real world event

Traffic Prediction - Kinematic



State pedestrian

$$\boldsymbol{X} = \begin{pmatrix} px \\ py \\ vx \\ vy \end{pmatrix}$$

Formal physical description of the object attributes

Model of pedestrian motion

deterministic

Stochastic

$$\boldsymbol{x_{k+1}} = \boxed{\boldsymbol{f}(\boldsymbol{x_k})} + \boxed{\boldsymbol{N}}$$

$$\begin{cases} px_{k+1} = px_k + vx_k * \Delta t + Nx \\ py_{k+1} = py_k + vy_k * \Delta t + Ny \\ vx_{k+1} = vx_k + Nvx \\ vy_{k+1} = vy_k + Nvy \end{cases}$$

Model is a physical approximation of real world event

Measurements (from sensor)

$$\boldsymbol{z_k} = \boxed{\boldsymbol{h}(\boldsymbol{x_k})} + \boxed{\boldsymbol{N}z}$$

Lidar

$$\boldsymbol{z_k} = \begin{pmatrix} px_k \\ py_k \end{pmatrix}$$

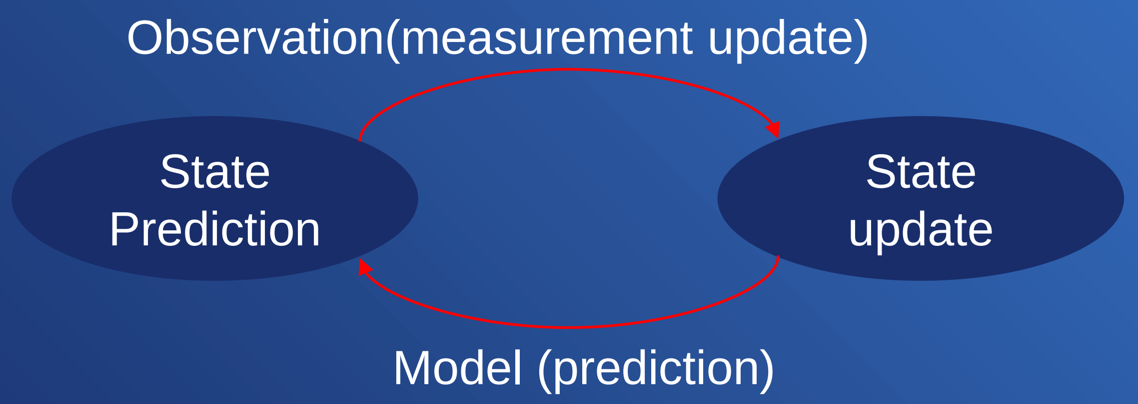
How I can adjust my prediction with the Information coming from the sensors?

BAYES FILTER

Object Tracking

- Object of tracking is to estimate objects state such location, speed and acceleration aver time.
- ADS need to track a lot of objects to maintain distance, predict trajectory and avoid them.
- Tracking is formulated as a sequential Bayesian filtering problem.
  - **Prediction Step**: Based on PRIOR time step predict current state based on model
  - **Correction Step**: Given predicted obj state at current time and sensors observation a POSTERIOR probability density of the state is calculated at current time

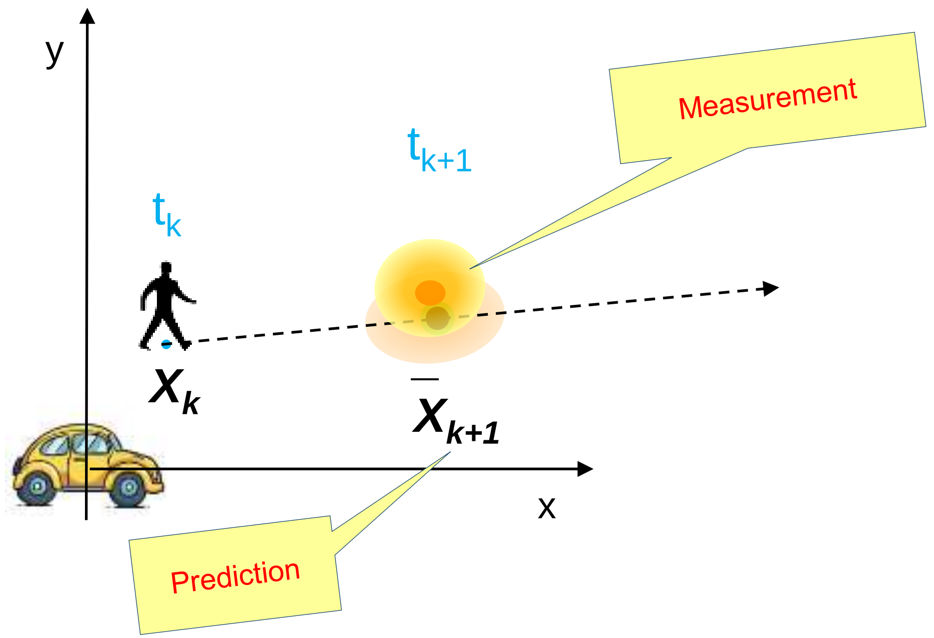
Use the information we have to predict the state until next observation arrives



Use new observation to correct our believe of the state

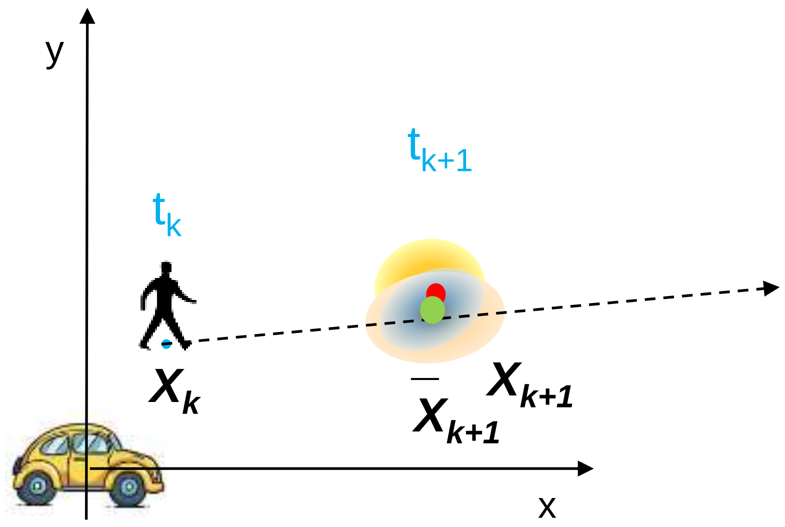
Object Tracking - BAYES FILTER

- Object of tracking is to estimate objects state such location, speed and acceleration aver time.
- Tracking is formulated as a sequential Bayesian filtering problem.
  - **Prediction Step**: Based on PRIOR state time step and model predict the current state



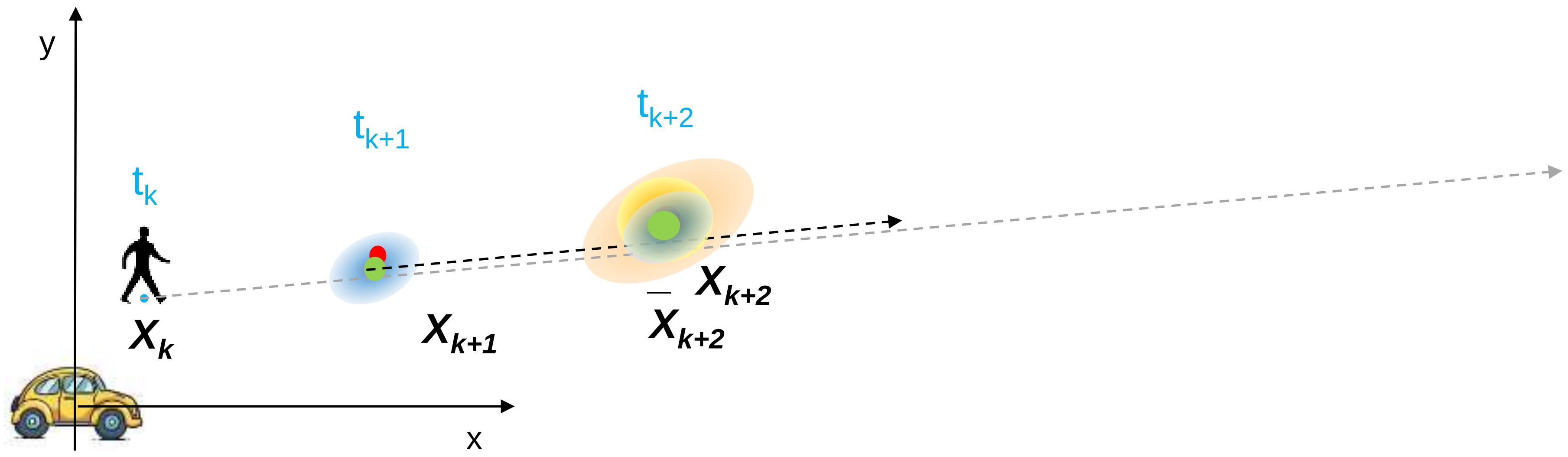
Object Tracking - BAYES FILTER

- Object of tracking is to estimate objects state such location, speed and acceleration aver time.
- Tracking is formulated as a sequential Bayesian filtering problem.
  - **Prediction Step**: Based on PRIOR state time step and model predict the current state
  - **Correction Step**: Given predicted obj state at current time and sensors observation a probability density of the state is calculated at current time (POSTERIOR)



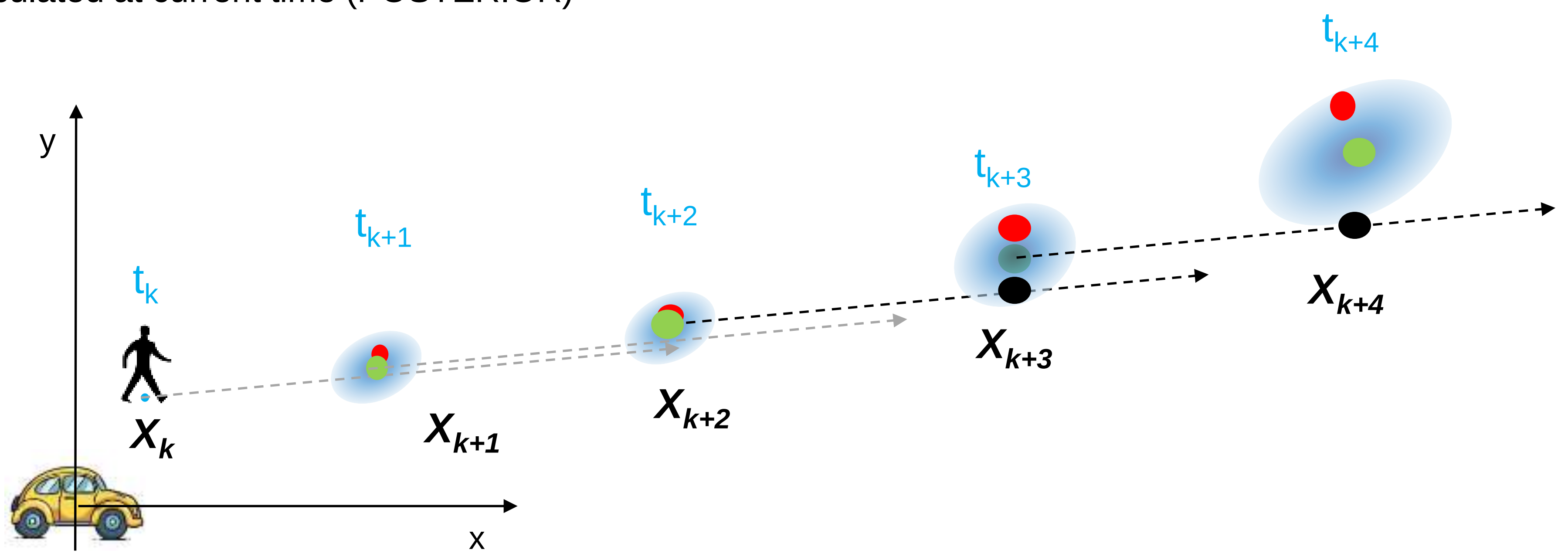
Object Tracking - BAYES FILTER

- Object of tracking is to estimate objects state such location, speed and acceleration aver time.
- Tracking is formulated as a sequential Bayesian filtering problem.
  - **Prediction Step**: Based on PRIOR state time step and model predict the current state
  - **Correction Step**: Given predicted obj state at current time and sensors observation a probability density of the state is calculated at current time (POSTERIOR)



Object Tracking - BAYES FILTER

- Object of tracking is to estimate objects state such location, speed and acceleration aver time.
- Tracking is formulated as a sequential Bayesian filtering problem.
  - **Prediction Step**: Based on PRIOR state time step and model predict the current state
  - **Correction Step**: Given predicted obj state at current time and sensors observation a probability density of the state is calculated at current time (POSTERIOR)

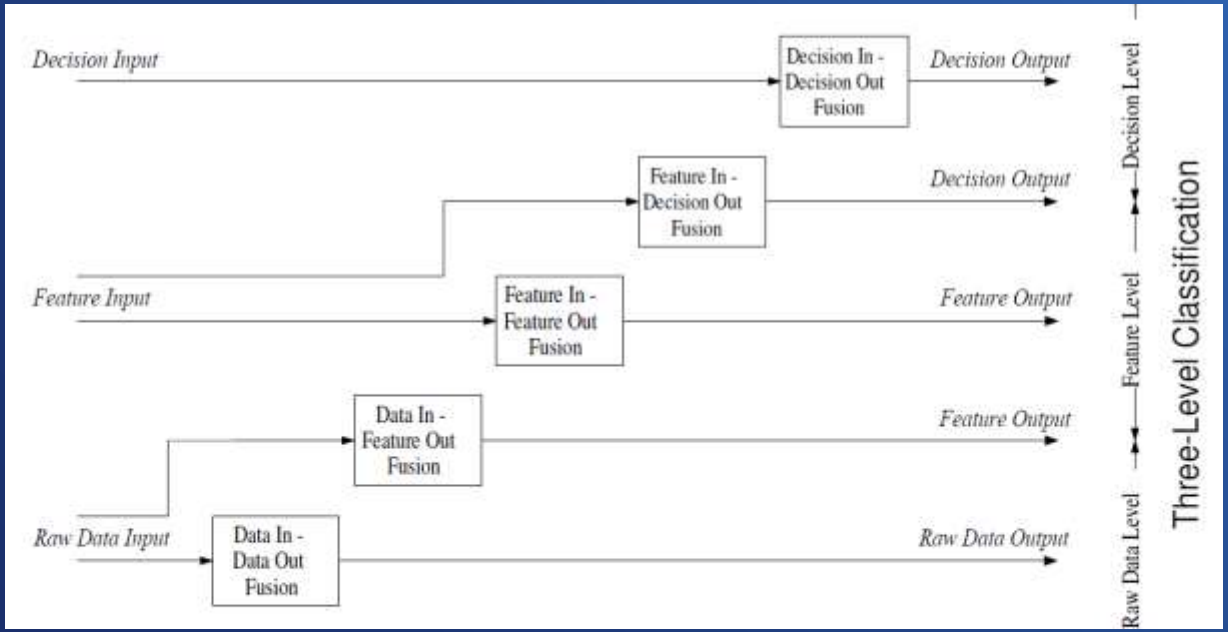


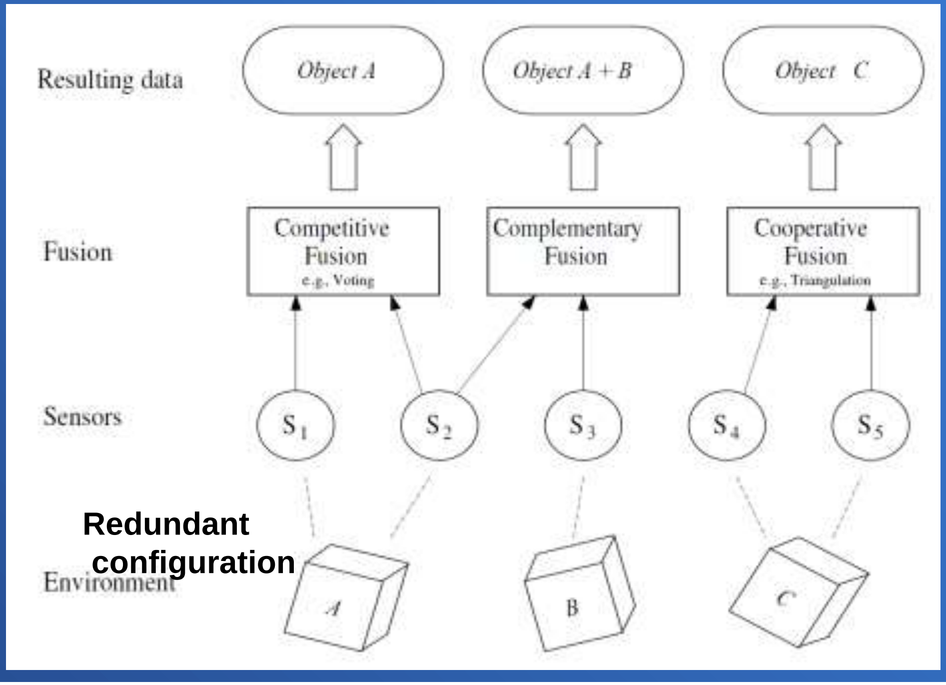
Bayes Filters Algorithms

ALGORITHM	Class	• Pros	• Cons
Kalman Filter & Extended Kalman Filter	Parametric – μ, Σ Gaussian Filters Unimodal Continous state state	• Well known. Easy to model Motion Update • Execute in Polynomial time	• Needs to Invert Matrix and uses Jacobian of parameters (tough already with 10-15 states)
Unscented Kalman Filter (UKF)	Parametric – μ, Σ Gaussian Filters Unimodal Continous state state	• For non linear systems the UKF produces equal or better results than the EKF. In many practical applications, the difference between EKF and UKF is negligible. • UKF does not requires Jacobian (derivative free)	• Same complexity of EKF, a little slower
Histogram Filters	Non-parametric. Split distribution in regions with constant values Discrete state state	• Can model multimodal distribution – Easier to compute	• Exponential time of execution.
Particle Filters	Montecarlo Methods – Samples from Posterior and associate Weight Continous state state	• Extremely easy to implement and can be adapted to all distribution • multimodal	• Needs large amount of samples (eg.1000) $\Rightarrow$ Computationally intense

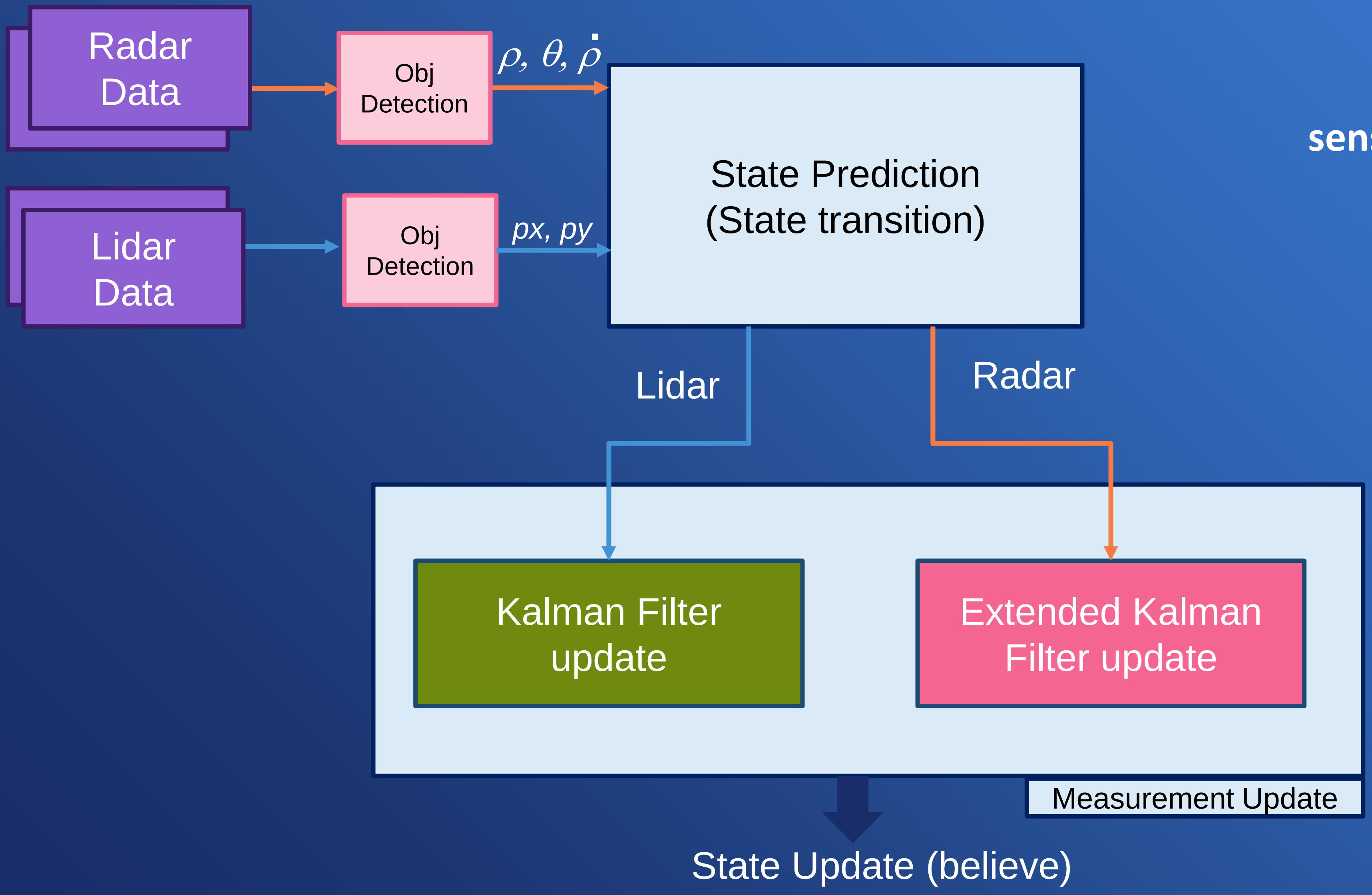
Sensor Fusion

- In order to achieve robustness and reliability we utilize sensor fusion strategy to combine multiple sensors together.
- Try to solve the uncertainty always present in the autonomous vehicle system:
  - Environment highly unpredictable
  - Sensors range, resolution and noise
  - Models inaccuracy
  - Limited Computation and real time requirements scarify accuracy





Tracking and sensor fusion

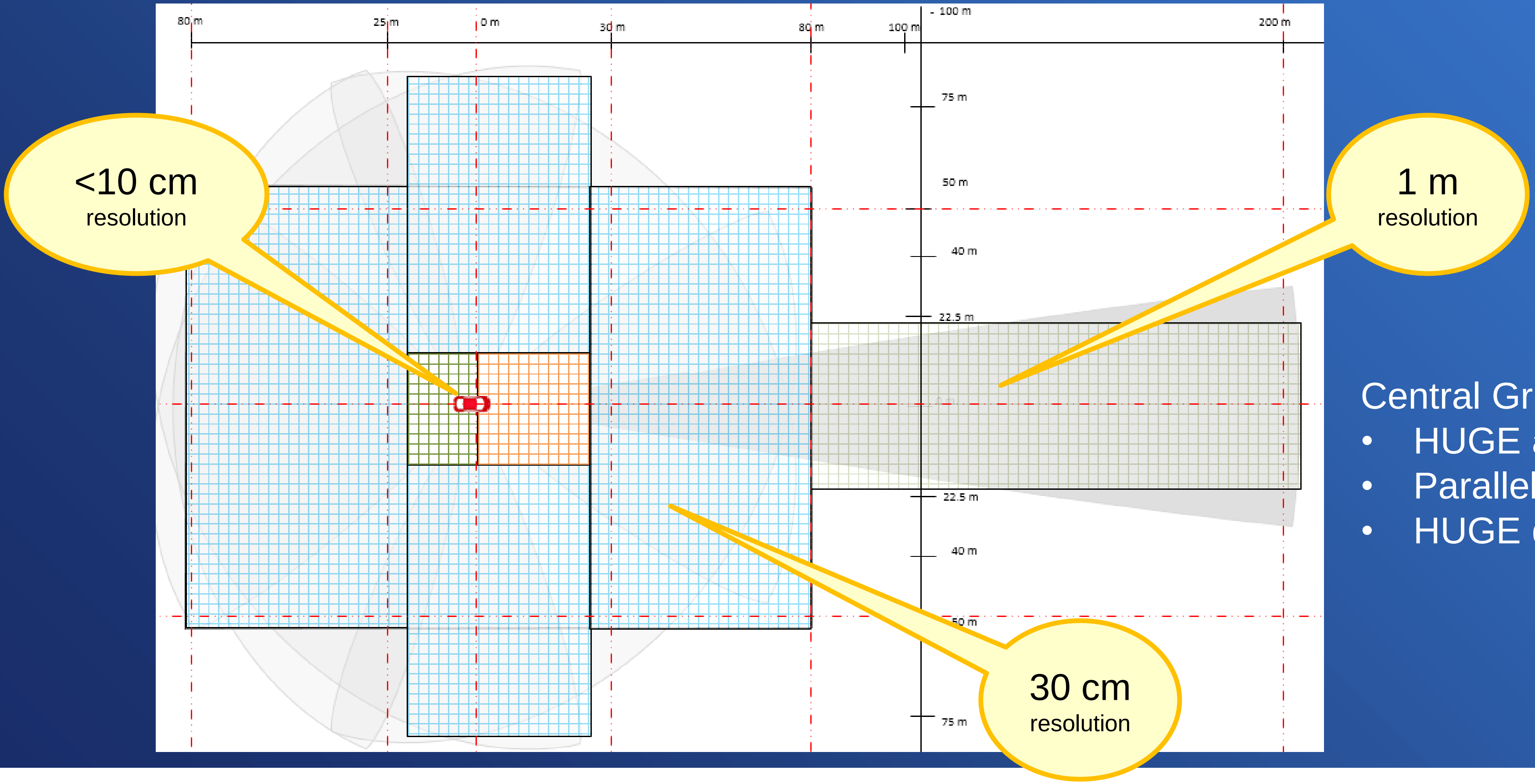


asynchronous sensors with different delays

The believe is updated asynchronously Each time the measurement is received regardless the source of the sensor.

Occupancy Maps

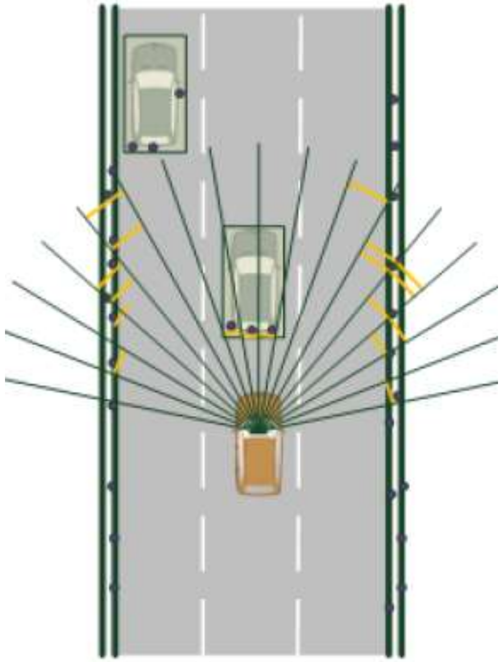
An occupancy map is a usually two-dimensional raster image uniformly distributed over the robot's working space. Each map pixel contains a binary value indicating whether the according space is free or occupied by an obstacle.

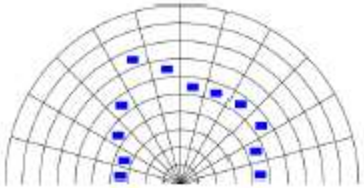


Central Grid computing

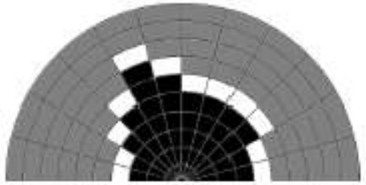
- HUGE amount of computation power
- Parallel Computation
- HUGE data structures (arrays) to be stored

Occupancy Grid Mapping example (single sensor)

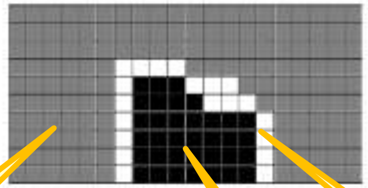




Sensor delivers a sparse free space representation



Compute a polar occupancy grid (dense representation) with the help of an inverse sensor model

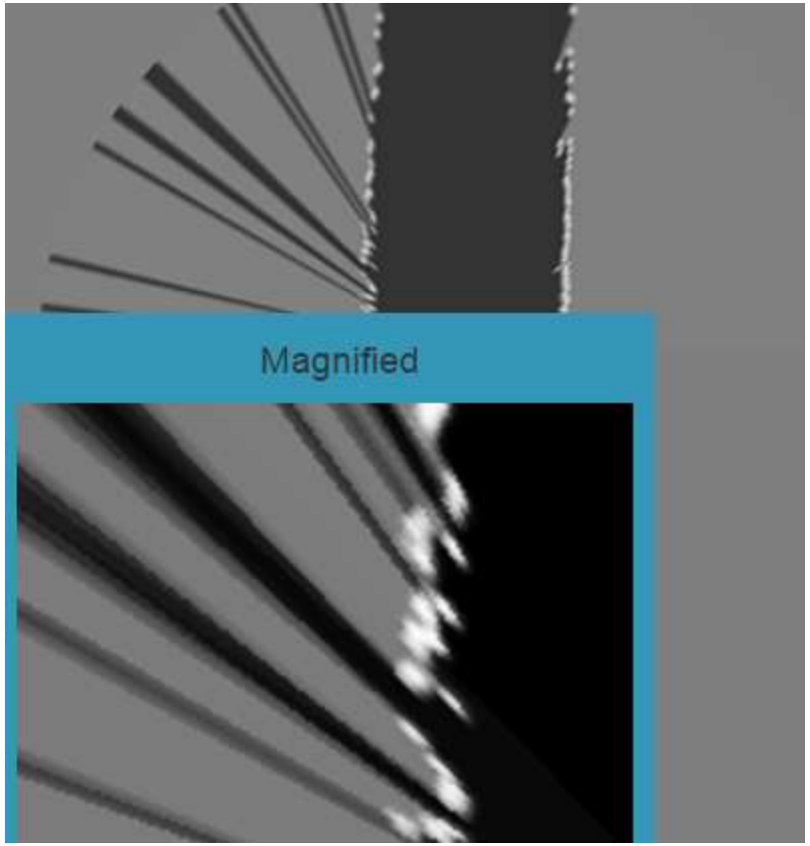


Transform from polar to Cartesian coordinates

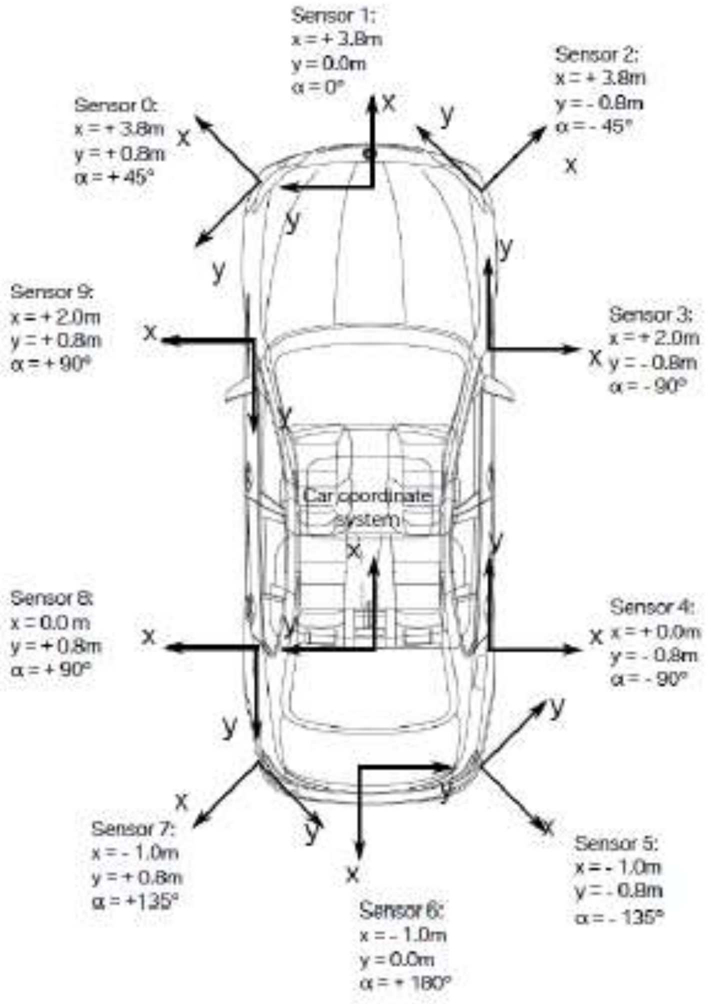
50% prob.

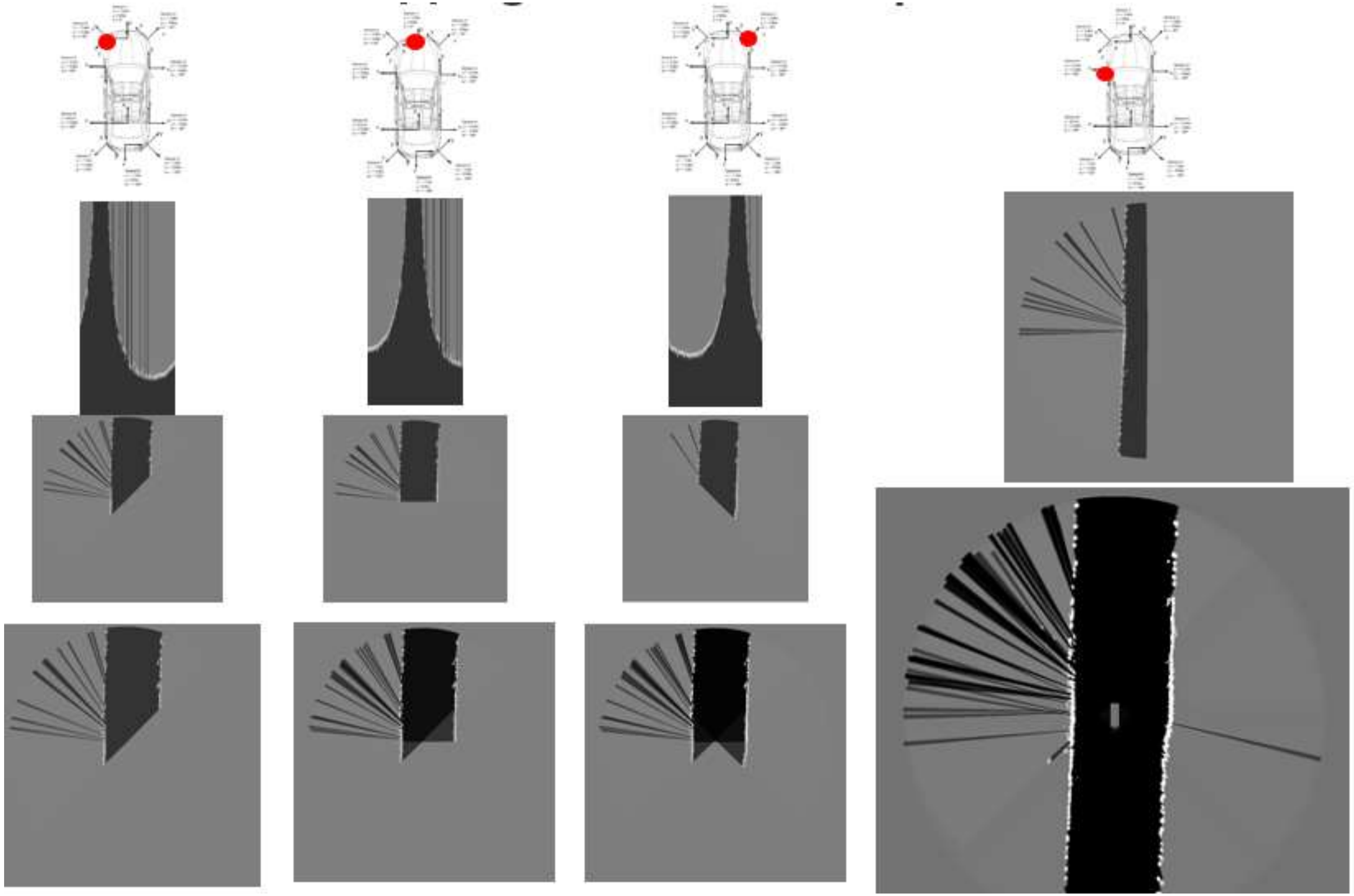
0% prob.

100% prob.



Occupancy Grid Mapping with Multiple sensors





Plan and Control

Routing → route

Route Planning (navigation)

At the highest level a route is planned through the road network.

Is the strategic target of the drive

Prediction → Objects trajectory

Traffic Prediction

Predict the behavior of the detected perception objects in the near future.

Output spatial-temporary trajectory points.

Behavior → decisions

Behavioral (local planning)

decides on a local driving task that progresses the car towards the destination and abides by rules of the road. Output high level semantic decision to be executed

Motion Planning → trajectory

Motion planning

selects a continuous path through the environment (trajectory) to accomplish a local navigational task.

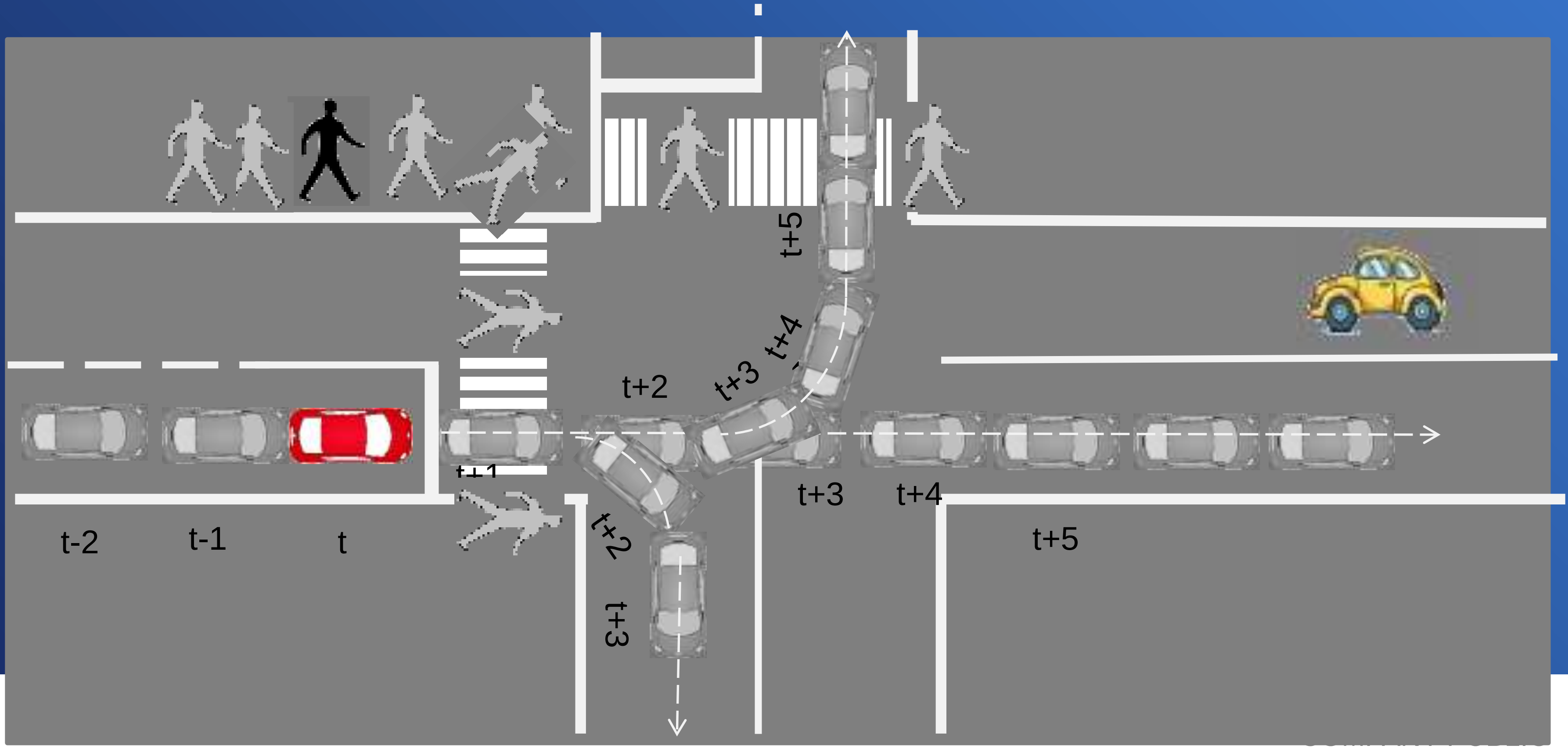
Feedback Control → control

(Feedback Control) Reactive Layer

A control system that reactively executes the vehicle planned trajectory.

Traffic Prediction

- The detected perception obstacles have attributes of position, velocity, headings, acceleration… Those are more on the kinematic side.
- The scope of traffic prediction is not just immediate prediction given the physical attributes but rather a behavioral level prediction spanning for periods of several seconds.
- Multiples factors must be considered like historical behavior, surrounding scenarios



Traffic Prediction

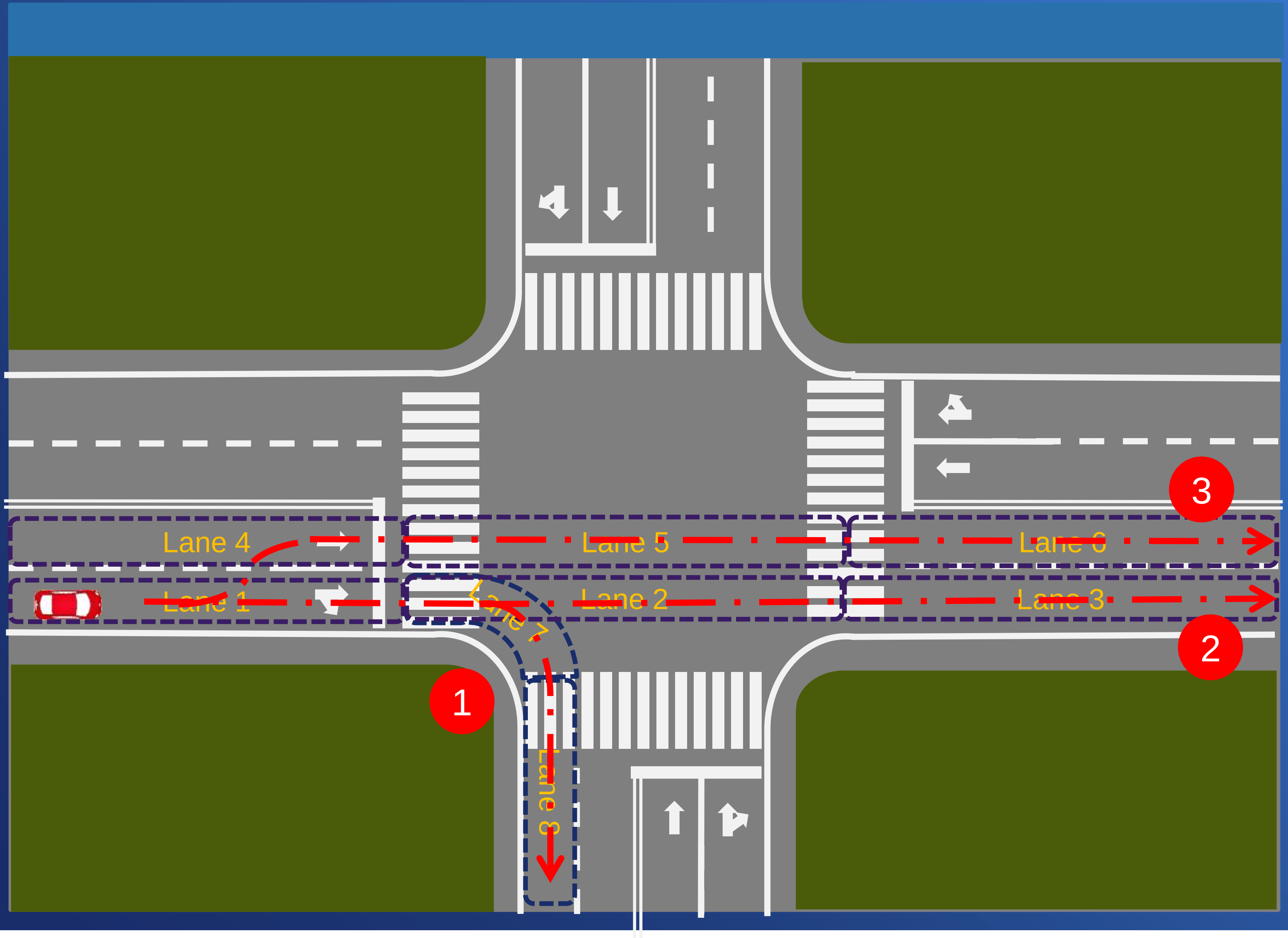
Prediction are mostly categorical and could also be formalized into classification problems and solved by machine learning.

It can be split into 2 sub-problems

- **Classification for road object behavior** : like change vehicle stay in lane, change lane or pedestrian cross at intersection and so on…
- **Regression problem** to associate the predicted path with speed and time info.

In reality the prediction is a very complex problem to solve because real maps, especially in urban environment, can be very complicated (multiple right/left, multiple lanes, and more than 4 way interceptions).

Traffic Prediction - Classification for road object behavior



1 Lane 1, Lane 7, Lane 8

2 Lane 1, Lane 2, Lane 3

3 Lane 1, Lane 4, Lane 5, Lane 6

We can label each trajectory as sequence of lanes.

We can formulate the behavioral prediction problem on the road into a binary classification on weather the vehicle will take a certain lane sequence.

Machine learning can be build considering:

- history features (w frames with absolute and lane relative position)
- Lane sequence features (sampled point in the lanes)
- Surrounding object features.

Traffic Prediction - Model for Vehicle Behavior Prediction

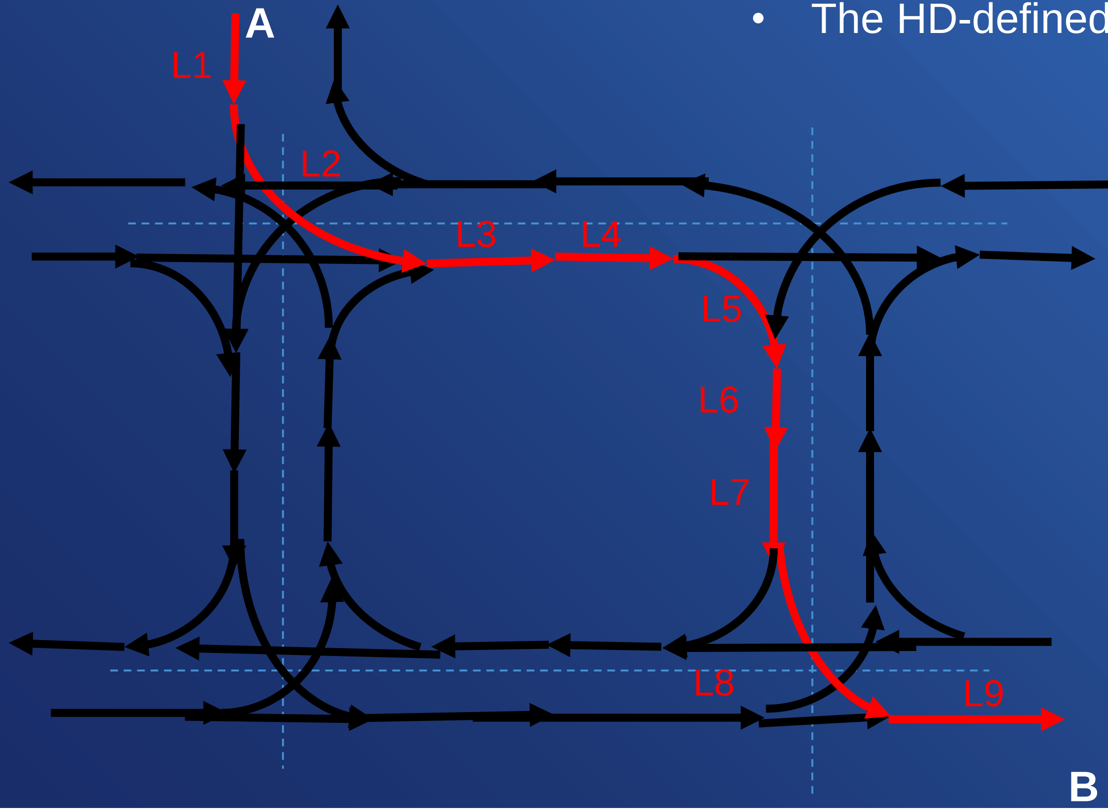
There are 2 class of ML models that can be used:

- **Memory less models** (SVM, DNN). The model stay the same once trained. Output does not depend on previous input. Easy to implement and train have the limitation of the history provided by feature. The predicted trajectory is maximum equal to the history (even if shorter is preferred). E.g. if the history is set to ~5 seconds, the max predicted trajectory must be <5s (better ~3).
- **Memory models** (Long Short-Term memory with Recursive NN) LSTN-RNN . The model have memory. Much more difficult to train. With RNN the history can be omitted because network will handle itself.

It the mapping and surrounding are not very complicated (like highway) then Memory less can work. For Urban driver cases, the RNN is preferred.

Lane Level Routing

- Solve the problem to go from “A” to “B” via a series of roads.
- Is different from standard Navigation. Its output is not from human but is for the other blocks
- (behavioral decision and motion planning)
- Requires a HD-MAP with indication of lanes
- The HD-defined lanes are not the natural lanes but correspond to the lane markers.



- L2, L5, L8 are the virtual turning lanes as defined by the MAP
- A long lane can be segmented in several lanes (L3,L4)

The output is a a lane segmentations defined in MAP:

$$L_i = \{ lane_i, start\_position_i, end\_position_i \}$$

Routing output from HD-MAP with indication of lanes

Lane Level Routing

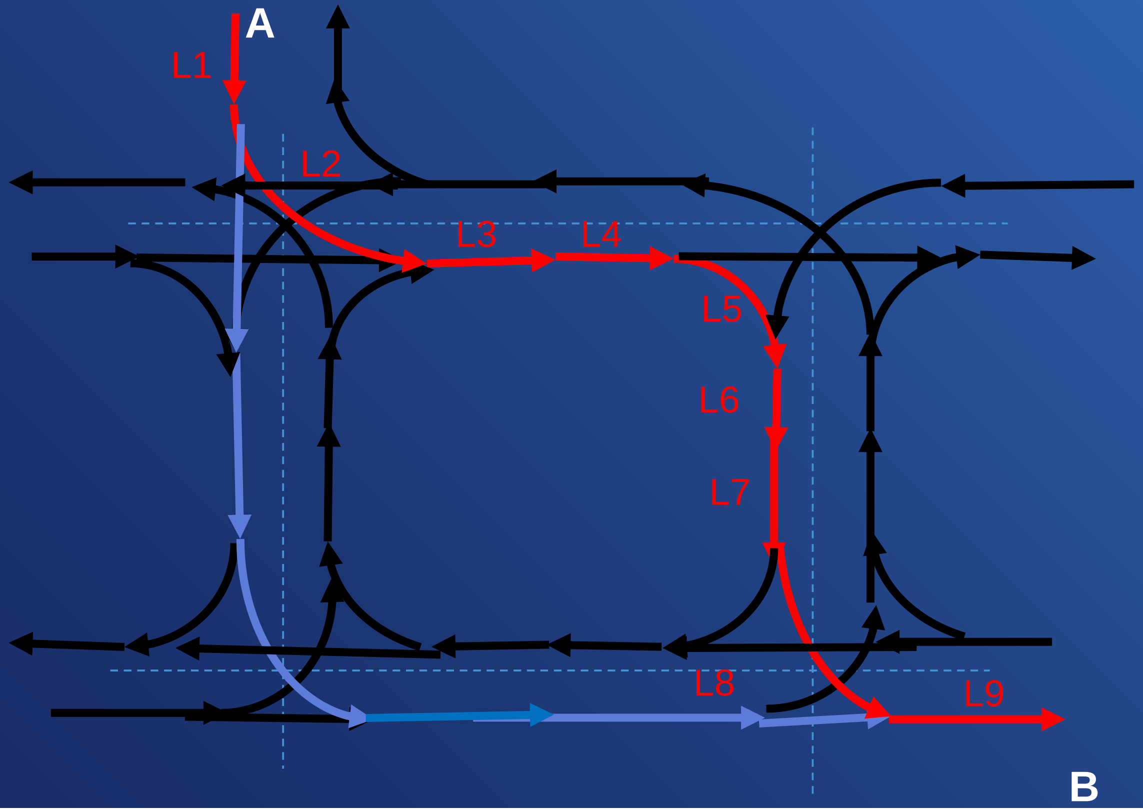
The ROAD GRAPH has been provided by the routing solving an <u>Optimal policy.</u>

$$L_i = \{ \text{lane}_i, \text{start\_position}_i, \text{end\_position}_i \}$$

Routing has to take into consideration the difficulties of certain movement. Difficulties of ADI are different from human, so the routing will be different.

Example of policy :

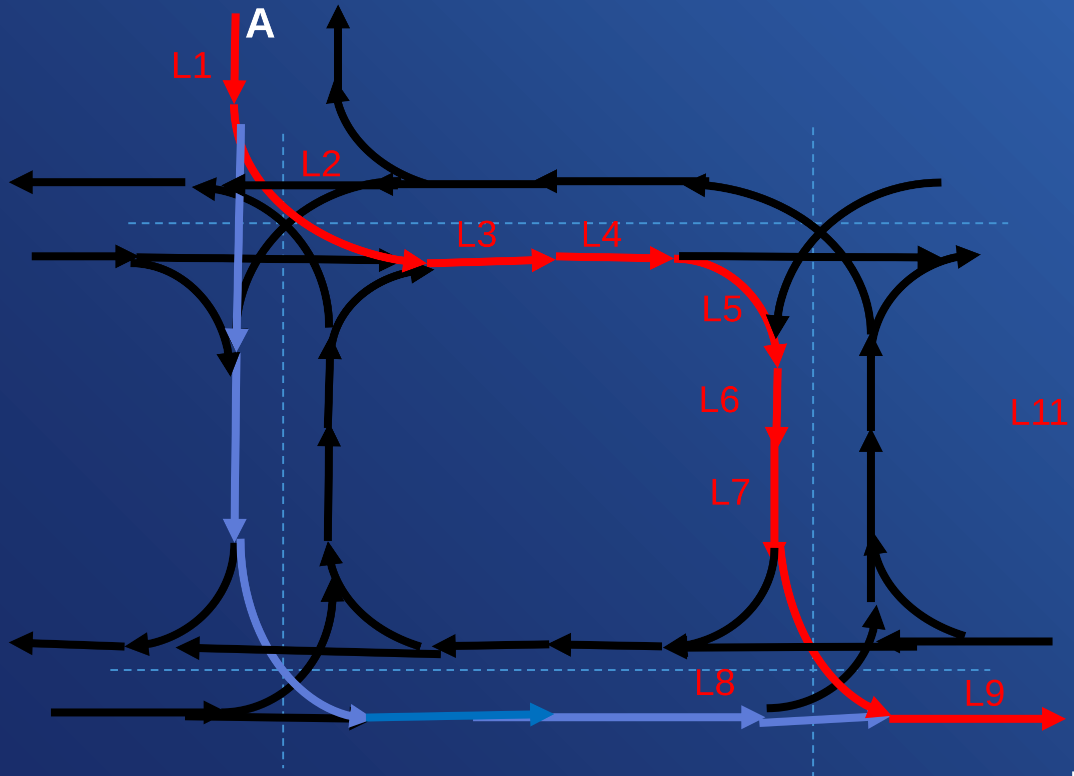
- Avoid to switch parallel lane if not really required (longer road, more time to fulfill, switching rad has an higher safety impact).
- Parallel lane switch will have higher cost.

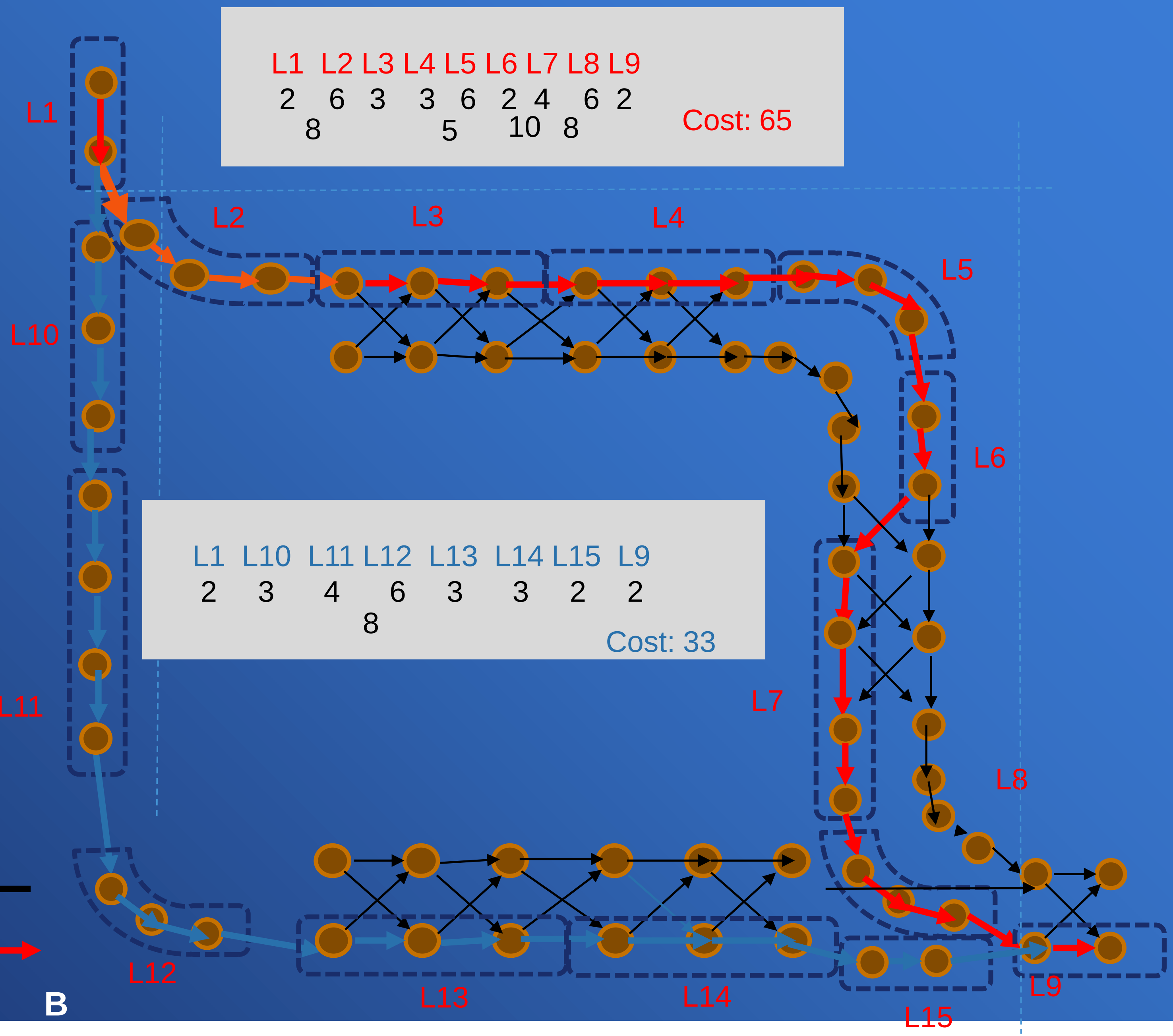


The ROUTING becomes a search of the shortest path into a weighted graph implementing an optimal policy.

Routing output from HD-MAP with indication of lanes

Lane Level Routing





Lane Level Routing

- ROUTING is solved thought weighted graph built from HD-MAPS
- Configuration of cost is the most important for authonomous drive
- Algorithm like Dijkstra, Greedy Best-First-Search, A*, are used to solve the optimal path

- Weights could changed based on other information (traffic) or dynamically adjusted
- There are 2 types of routing request:
    - Start the journey (from the driver)
    - Recalculate (from the other blocks). E.G. if the dynamic traffic oblige to change a lane then routing must be recalculating.
        - STRONG ROUTING: downstream follow strongly the routing. Decision an planning will follow the lane by lane set from routing (L1/L2)
        - WEAK ROUTING: the decision could not follow the routing under certain conditions (vehicle behavior).
- SAFETY FIRST WILL BE the main DRIVE for any policy.

Behavioral Decision

Act as a Co-Pilot in the autonomous vehicle Motion Planning and Control

ROUTING
sequence of lanes with start and end

AD VEHICLE ATTRIBUTES
Position, current lane, relative position to lane, speed, Heading, target lane

HYSTORICAL VEHICLE information
The previous frame or cycles of behavior decisions

OBSTACLE INFORMATION
All the objects within a certain radius. Each obj has located lane, speed, heading and predicted trajectory.

TRAFFIC AND MAP
The lanes and relationships as defined by HD MAP. Info also on traffic lights, stop signs and pedestrian cross-walk (from MAPPING and/or PERCEPTION)

LOCAL TRAFFIC RULES
City speed limits, legal to make right turn on red TL

SAFETY FIRST

Behavior

decision

CRUISE
- Current Lane
- Speed Limit in the lane

FOLLOW
- Current Lane
- Id of vehicle to follow
- Speed of current lane and vehicle to follow
- Min distance to keep

TURN
- Current Lane
- Target lane
- Left / Right
- Speed limit for turning

CHANGE LANE
- Current Lane
- Target lane
- Overtaking and speed up 10m/s
- By yielding and slow 2m/s

STOP
- Current Lane
- Id of obj to stop, if any
- Stop by 1m behind the obj to stop

Behavioral Decision

- The approach is based on a Divide and Conquer idea, to decompose the surrounding environment into layers and solve them individually

- The main stream solutions are RULES BASED deterministic models, mostly based on Markov Decision Process. A MDP is defined by 5 elements (S,A,P_a,R_a,γ):
  - S: state space, represented with grid map and map elements.
  - A: behavior decision output space (Cruise, Follow, Turn, Change Lane, Stop)
  - $P_a(s,s') = P_a(s', s|a)$ Prob. To reach s' being on s anc taking the action s.
  - $R_a(s,s')$ reward function of transition from s to s' taking action a. Factors like safety, confort, execution of rout planning needs to be considered.
  - γ Decay factor for reward. Guarantee that present is more valuable than future reward.

Scenario Based Divide and Conquer

SYNTHETIC DECISION

CRUISE
Current Lane
Speed Limit in the lane

FOLLOW
Current Lane
Id of vehicle to follow
Speed of current lane and vehicle to follow
Min distance to keep

TURN
Current Lane
Target lane
Left / Right
Speed limit for turning

CHANGE LANE
Current Lane
Target lane
Overtaking and speed up 10m/s
By yielding and slow 2m/s

STOP
Current Lane
Id of obj to stop, if any
Stop by 1m behind the obj to stop

SYNTETHIC DECISION is about how the AV should behaves considering all the information available, including all the road objects

INDIVIDUAL DECISION only relates to individual elements in the surrounding world. The SYNTETIC DECISION will be calculated considering a consolidation of the individual decision.

INDIVIDUAL DECISION

VEHICLES

FOLLOW
- Id of vehicle
- Speed and min distance

STOP
- Id of vehicle
- Distance of the vehicle to stop

Attention
- Id of vehicle
- Min distance

Overtake
- Id of vehicle
- Min distance for overtaking
- Min time gap for overtaking

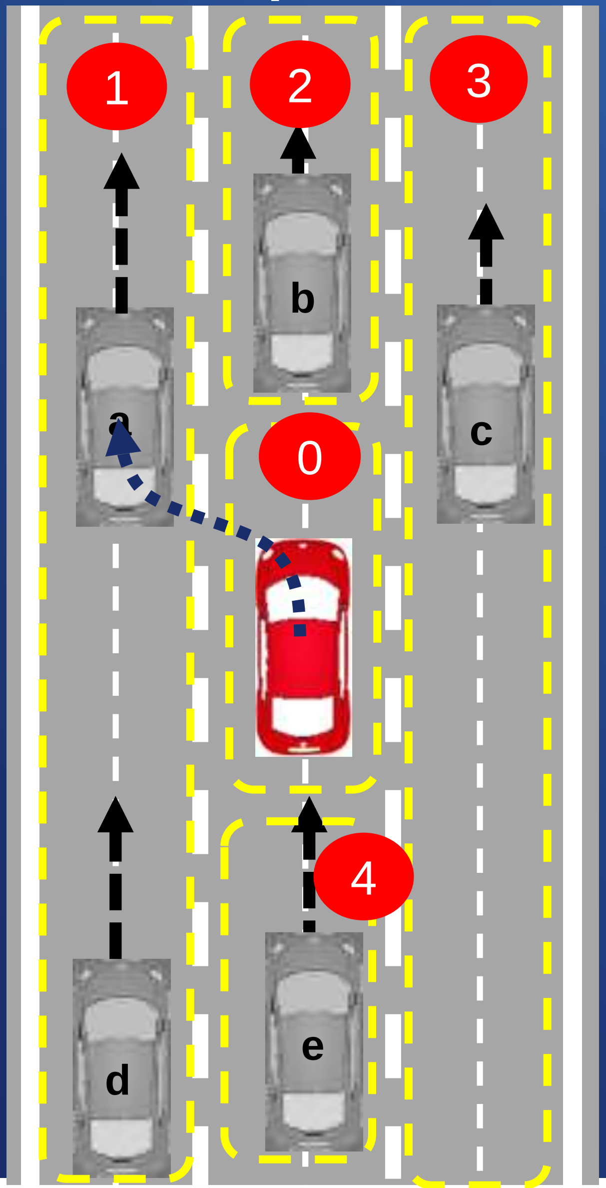
YIELD
- Id of vehicle
- Min distance for overtaking
- Min time gap for overtaking

Pedestrian

STOP
- Id of vehicle
- Distance of the ped. to stop

SWERWE
- Id of vehicle
- Distance of the ped. to Swerve

Examples



Scenario and Individual Decision
0: AD VEHICLE

1: Left lane
Overtake d
Yield a

2: Front lane
Attention b

3: Right lane
Ignore c

4: Back lane
Ignore e

Synthetic Decision
Switch lane from current to left; yield a, overtake d, attention to b on current lane

Layered Scenario and Decisions

Layer 0
- Master Vehicle

Layer 1
- Left Lane
- Front Vehicle
- Right Lane
- Crosswalk
- Traffic Lights

Layer 2
- 4-way Intersection
- 3-way Intersection
- Lane Split
- Lane Merge

↓

Merge individual decisions to Guarantee safety

↓

Synthetic Decision

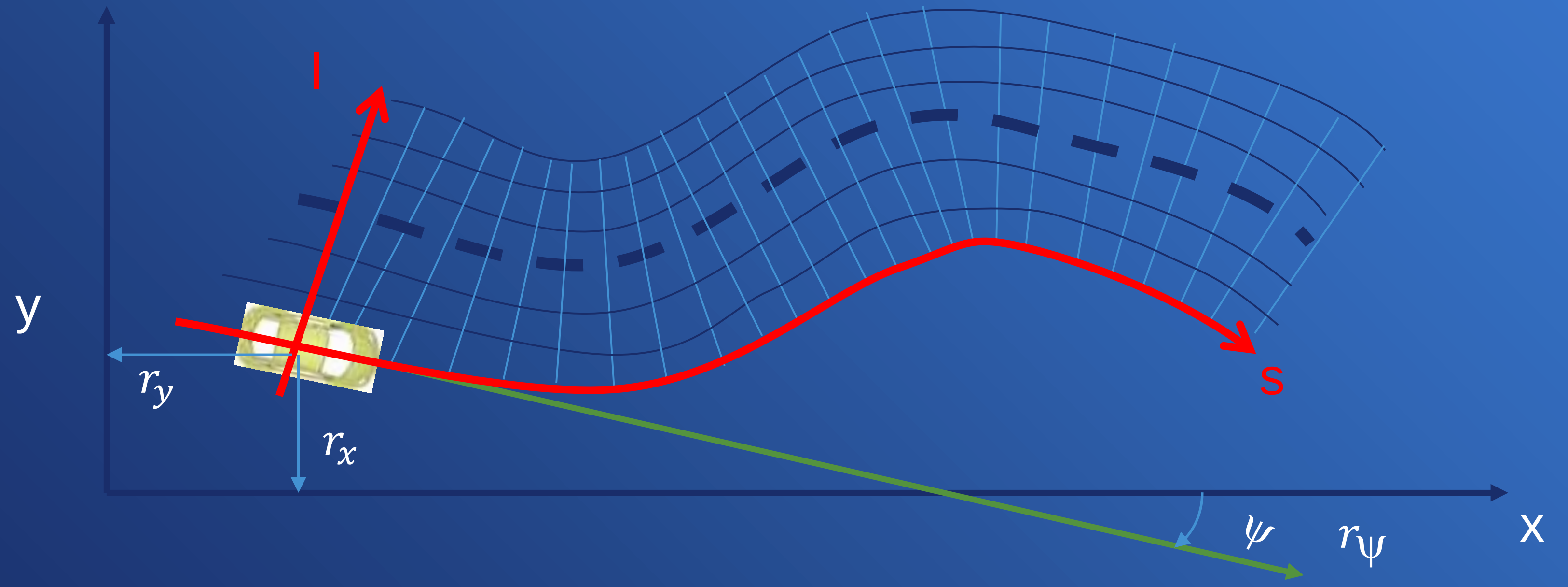
Motion Planning

- The task is to generate a trajectory and send it to feedback control for physical execution. Optimize the spatial-temporal trajectory among the points.
- The planning is 2D problem with limited degrees (break, throttle and steer) so not too complex.
- The output trajectory is specified as a sequence of points. Each point contain attributes like position, time, speed, curvature, higher order derivative of curvature.
- Optimization is usually represented with cost of different solutions. The goal is to search from min solutions. COST is build to obey to the Behavioral module.
- The solution again is to split the problem:
  - Solve the path planning without speed info
  - Solve the speed planning.

The computed trajectory must be harmonious (and the car is not an harmonious system) the solution must have property of spline trajectory.

Motion Planning

We introduce a SL-coordinate system. (s – longitudinal, l lateral)



Vehicle pose)

$$x = \begin{pmatrix} px \\ py \\ v \\ \psi \\ \kappa \end{pmatrix} \qquad \dot{\psi} = \kappa\psi$$

Reference line

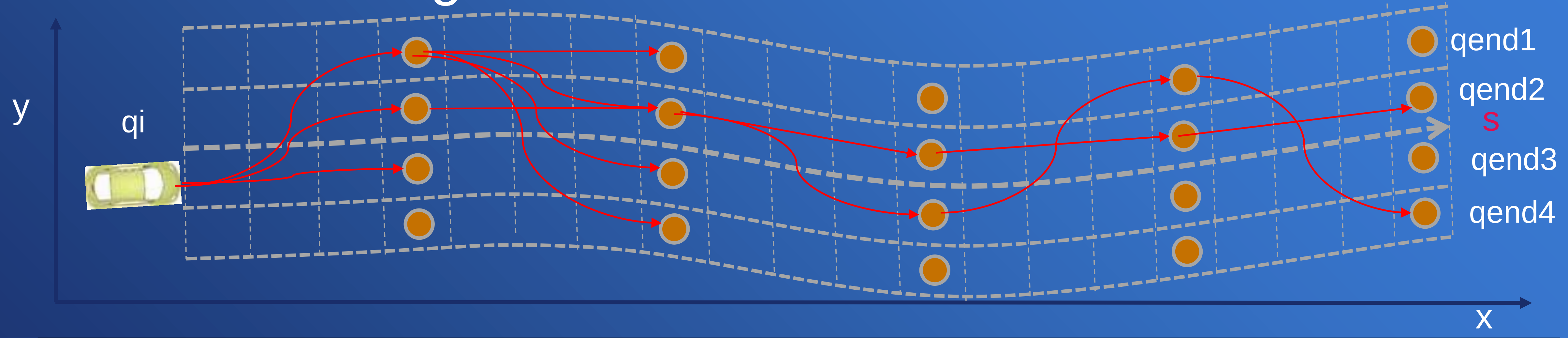
$r(s) = [r_x(s), r_y(s), r_\psi(s), r_\kappa(s)]$

$$x_r(s,l) = r_x(s) + l\cos(r_\psi(s) + \frac{\pi}{2})$$

$$y_r(s,l) = r_y(s) + l\sin(r_\psi(s) + \frac{\pi}{2})$$

$$\psi_r(s,l) = r_\psi(s)$$

Motion Planning



Split s and l and distribute 4 points on l and 5 points on s

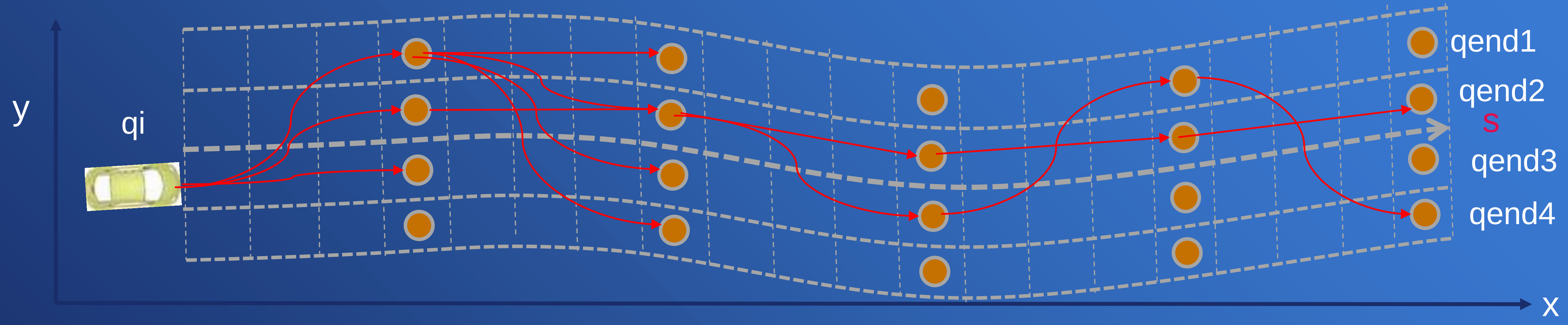
The total number of path is 4^5=1024 path among which the path planning, given the Behavior constrain, need to search for optimal cost solution.

The path have a cubic or quintic spline $\kappa(s) = \kappa_0 + \kappa 1\ s + \kappa 2\ s^2 + \kappa 3\ s^3 + \kappa 4\ s^4 + \kappa 5\ s^5$ the second order (wheel rotating speed) is not continuous in cubic. For low speed is ok but for high speed quintic spline must be implemented.

Furthermore there are other dependency:

$\kappa_0 = \kappa 1 = \frac{d\kappa(0)}{ds}$ $\kappa 2 = \frac{d^2\kappa(0)}{d^2 s}$ so to connect qi=$[xi, yi, \psi\ i, \kappa\ i]$ to qend=$[xe, ye, \psi e, \kappa e]$ so the free parameter are $(\kappa e, \kappa 3, \kappa 4, \kappa 5)$ that can be calculated with gradient descent.

Motion Planning



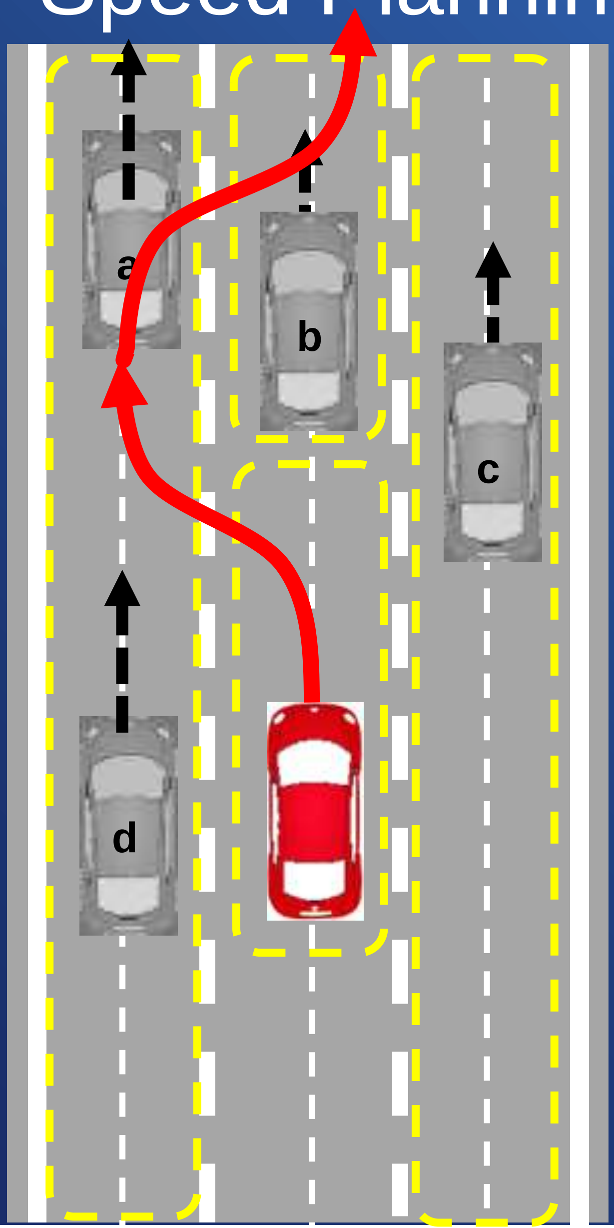
MOTION PLANNING only consider s; l will be considered in **SPEED PLANNING**
COST function:

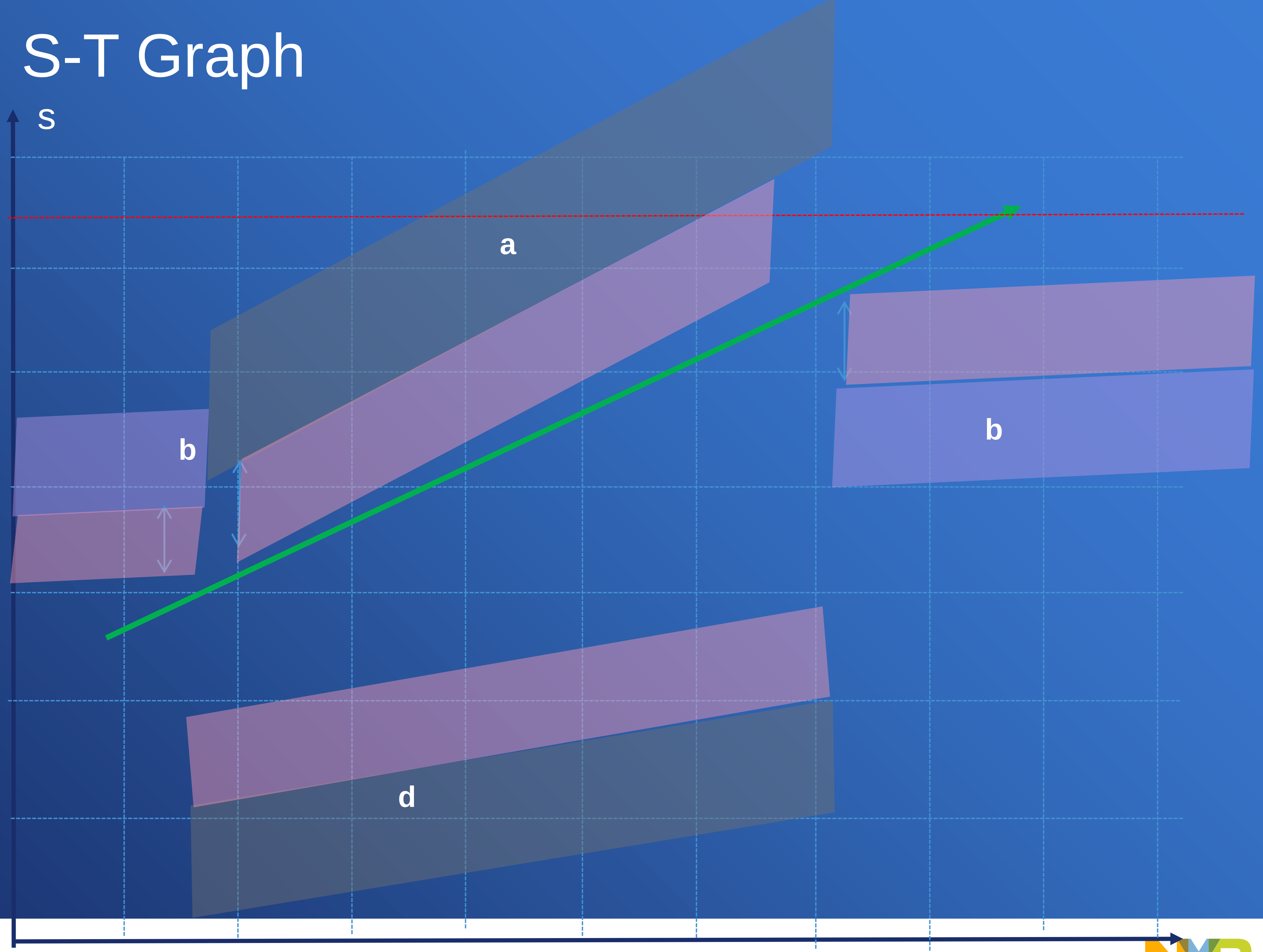
- **Road-map related aspects**: Path must be closed to central reference line of the lane based on Behavior (e.g. if FOLLOW, then cost will be related to lateral distance)

- **Obstacle related:** collision to STATIC objects must be penalized. Motion planning cannot address moving obj that will be addressed in SPEED PLANNING

- **Comfort and control feasibility:** Path must be smooth, (expressed by derivative). Not only the path but the also the connection between two path.

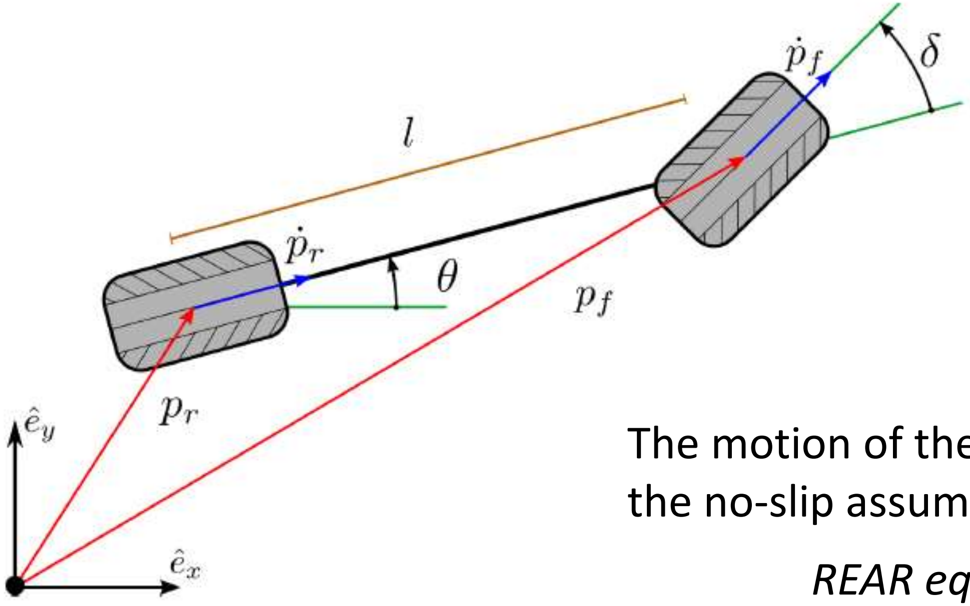
Speed Planning S-T Graph





Overtake b, Overtake d
Yield a

The Bicycle Model



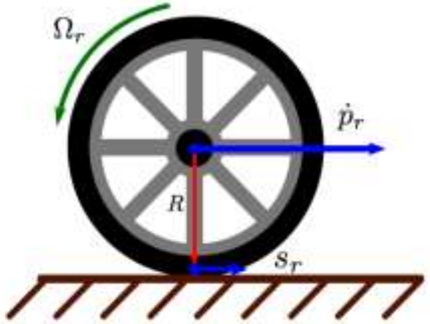
- the vectors p_r and p_f denote the location of the rear and front wheels in a stationary or inertial coordinate system with basis vectors ($\hat{e}_x$, $\hat{e}_y$, $\hat{e}_z$).
- θ is an angle describing the direction that the vehicle is facing (vehicle heading angle). This is defined as the angle between vectors $\hat{e}_x$ and pf − pr.
- δ is the steering angle of the front wheel.

The motion of the points p_r and p_f must be collinear with the wheel orientation to satisfy the no-slip assumption.

REAR equation: $$(\dot{p}_r \cdot \hat{e}_y)\cos(\theta) - (\dot{p}_r \cdot \hat{e}_x)\sin(\theta) = 0,$$

Front equation: $$(\dot{p}_f \cdot \hat{e}_y)\cos(\theta+\delta) - (\dot{p}_f \cdot \hat{e}_x)\sin(\theta+\delta) = 0$$

In general, sr and ṗr are not collinear and may have nonzero components normal to the plane depicted.



rear wheel (similar for front wheel)

At speed > 50km/h a more complex model considering the wheel drift model must be considered.

SAFE AND SECURE MOBILITY

AT THE HEART OF AUTONOMOUS DRIVING

NXP

SECURE CONNECTIONS
FOR A SMARTER WORLD

www.nxp.com