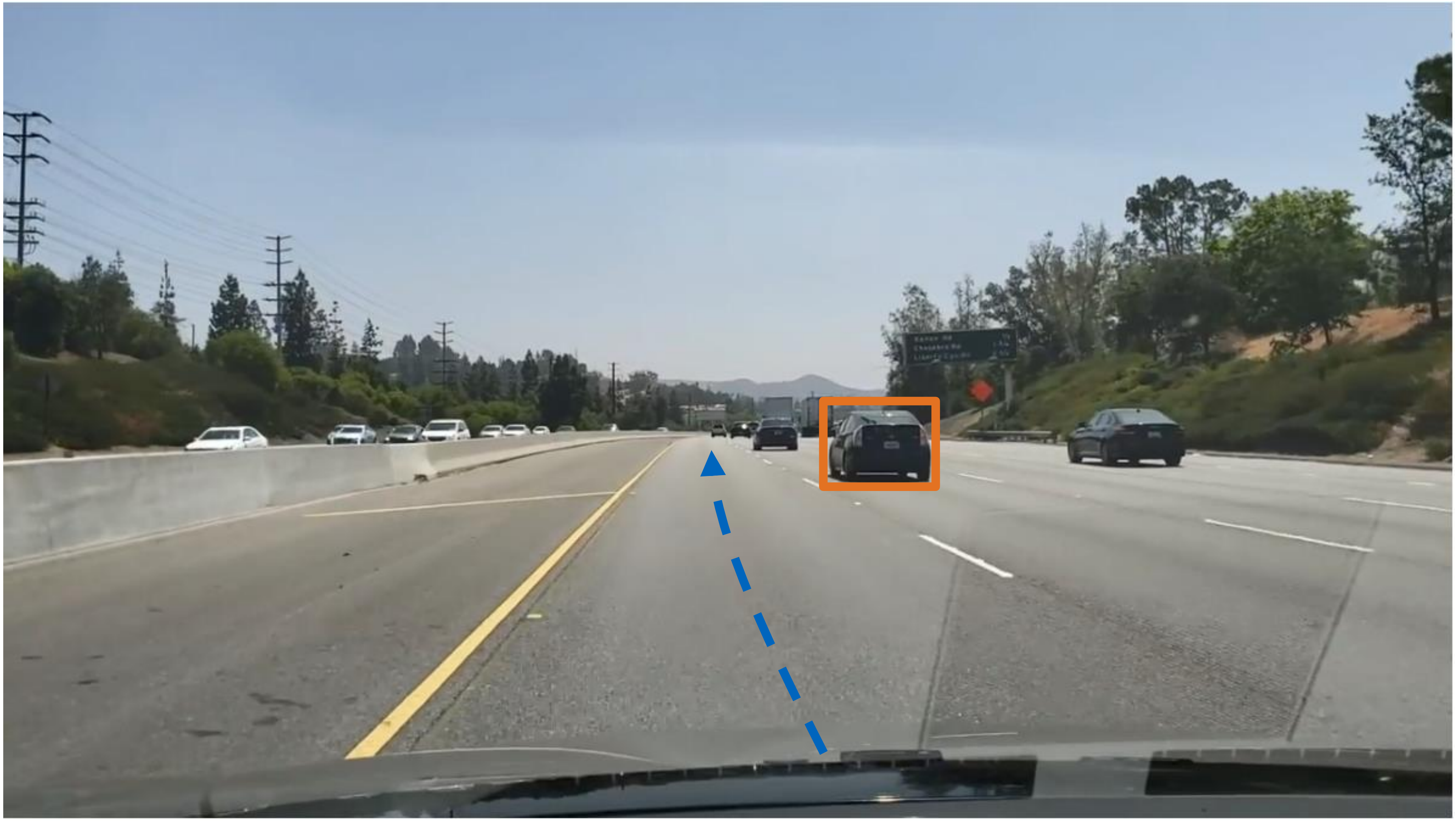
Institute of Automotive Technology
Faculty for Mechanical Engineering
Technical University of Munich

Autonomous Driving Software Engineering

Prof. Dr.-Ing. Markus Lienkamp

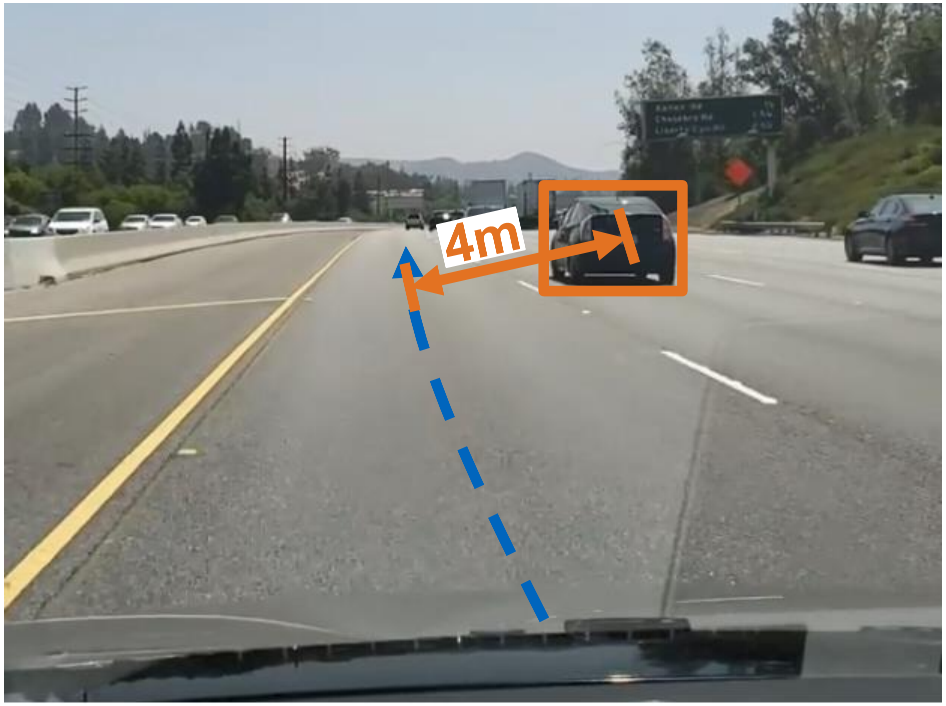
Nico Uhlemann, Dipl.-Ing.

Simon Sagmeister, M. Sc.



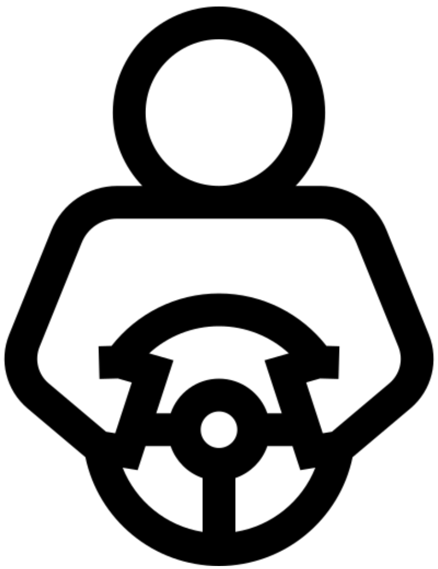
https://www.youtube.com/watch?v=01_SVshk-MY

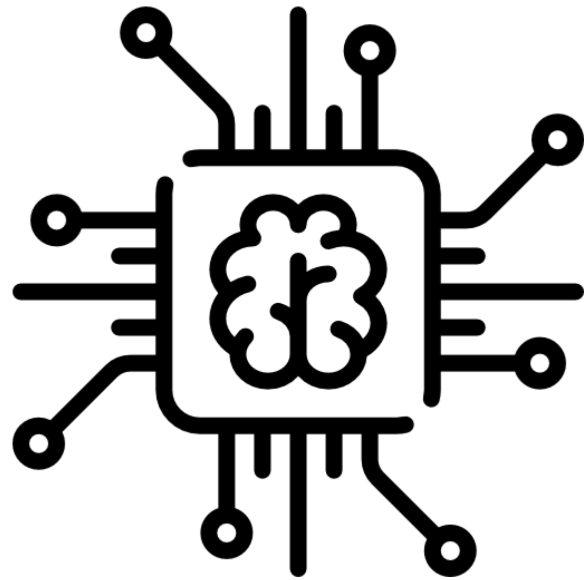
https://www.youtube.com/watch?v=SBpUppJACxA


4m


15m

How to anticipate other traffic participants?

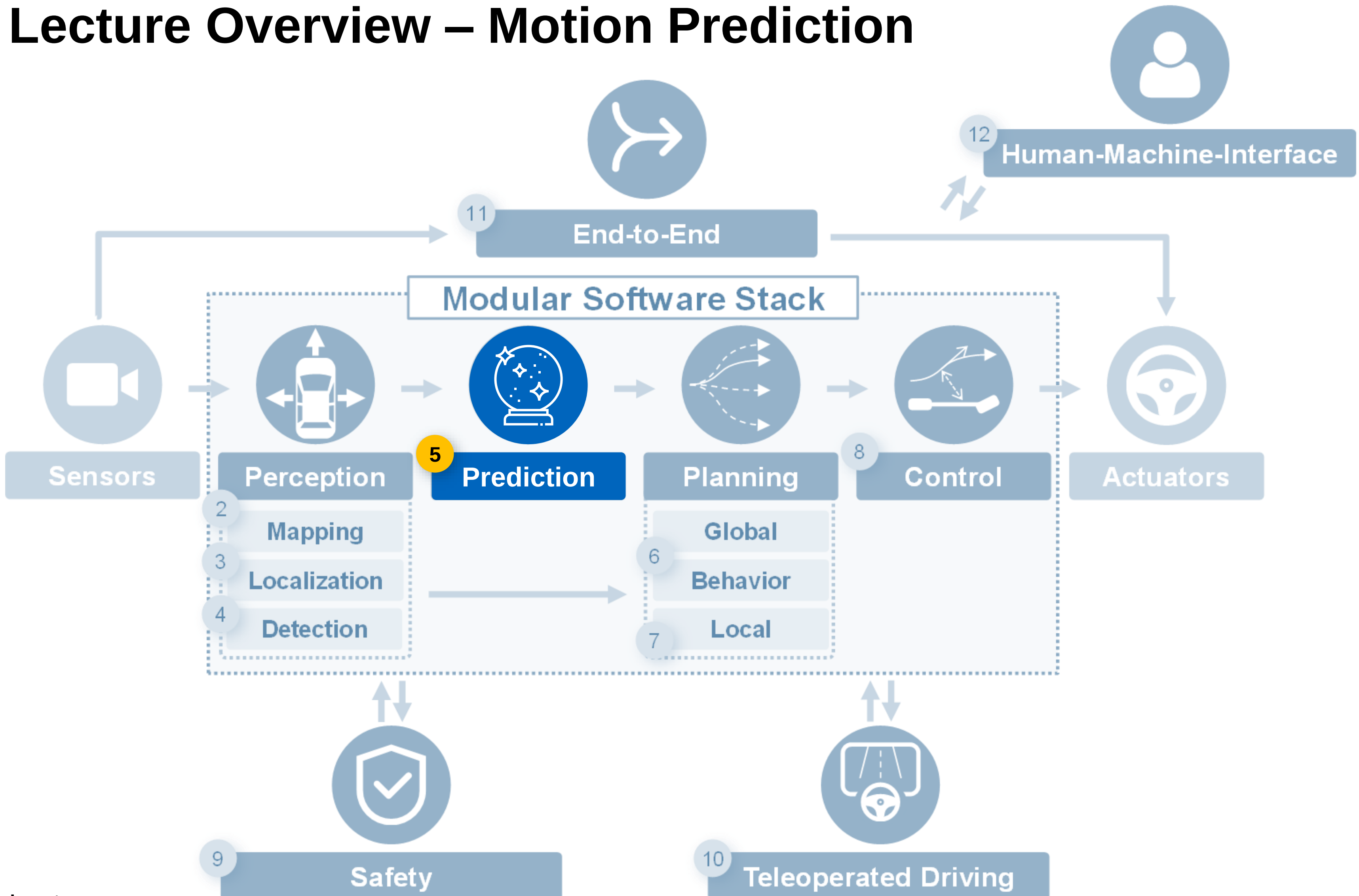




Anticipatory Driving through Riding Experience

Motion Prediction through Scenario Understanding

Lecture Overview – Motion Prediction



X = Lectures

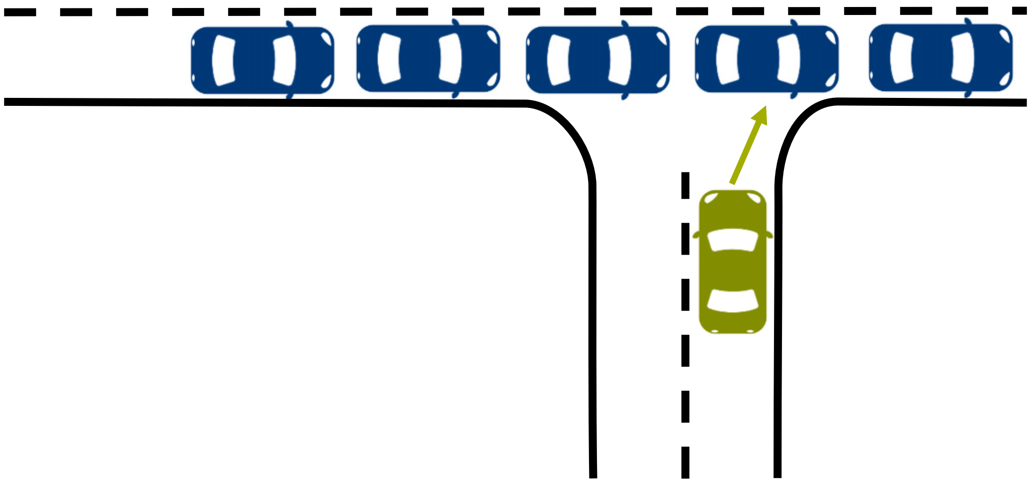
Prediction
Prof. Dr. Markus Lienkamp

Dipl.-Ing. Nico Uhlemann

Agenda

1. **Foundations**
2. Knowledge-Based Prediction
   a. State Estimation
   b. Reachable Sets
3. Learning-Based Prediction
   a. Clustering and Classification
   b. Deep Learning
   c. Reinforcement Learning
4. Summary and Outlook

Foundations – Why is prediction needed?



„Sense – Plan – Act“ works in static environments, but falls short in complex and dynamic situations like road traffic

Environments

Sense

↓

Plan

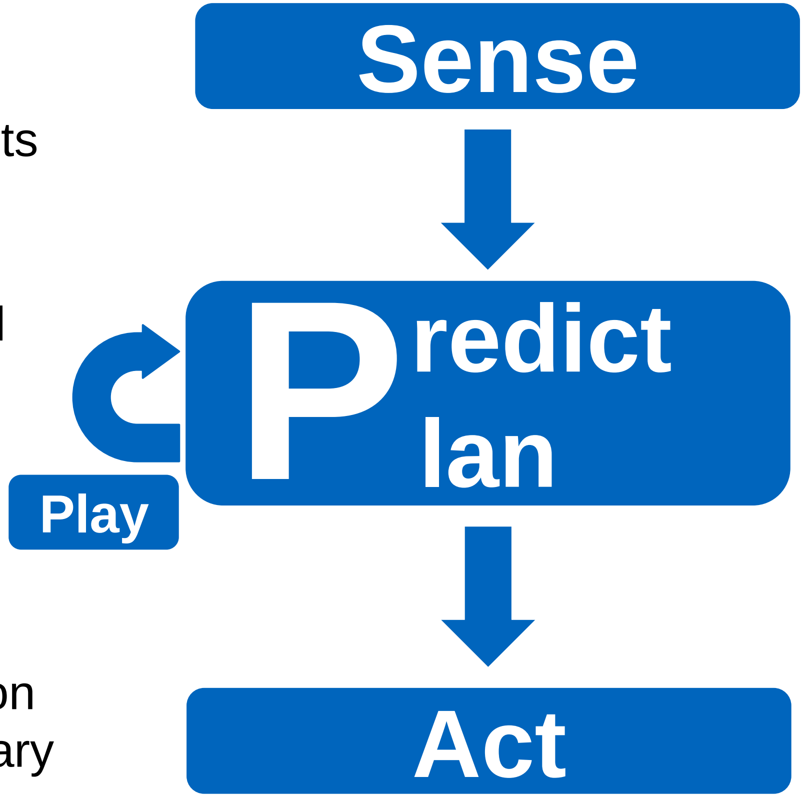
↓

Act

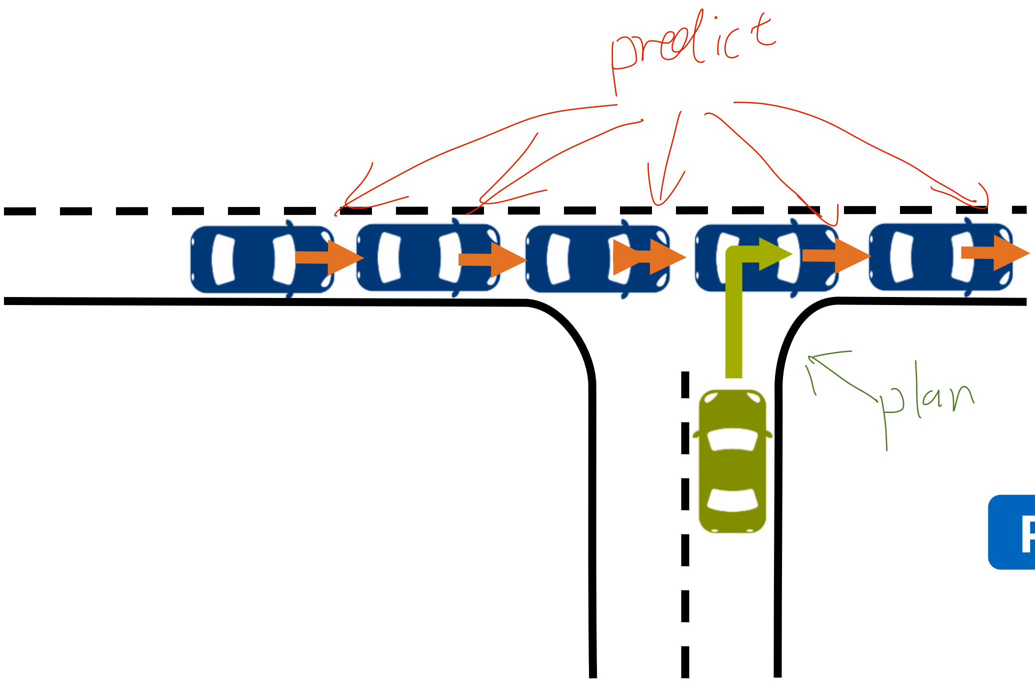
Foundations – Why is prediction needed?

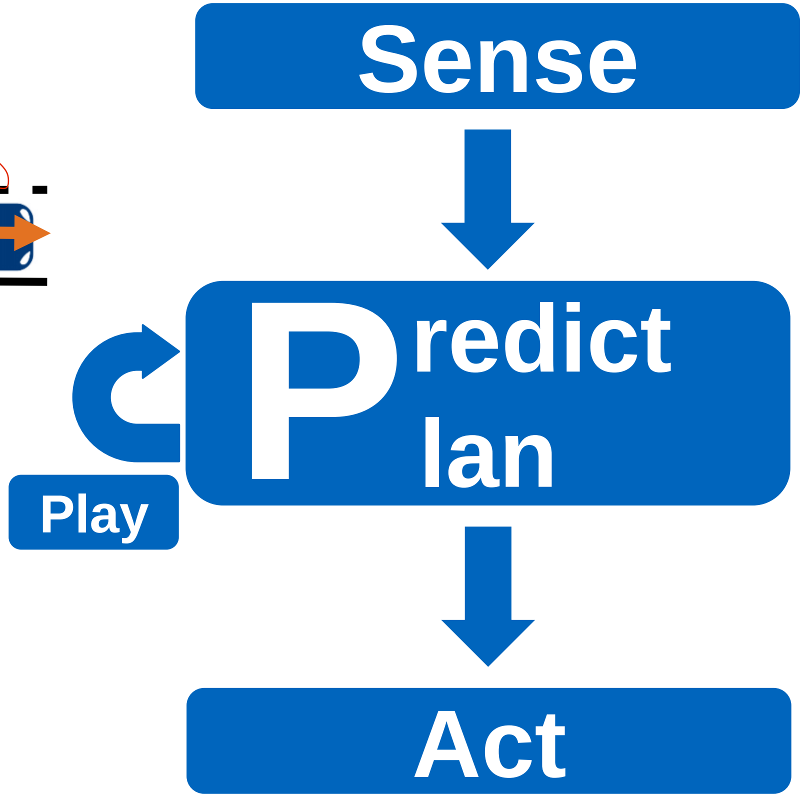
We need to:

- **Understand**
  Determine intentions of surrounding objects and reason about possible behavior
- **Predict**
  Forecast what the surrounding objects will do and quantify the probabilities involved
- **Plan**
  Evaluate the collision probability of the possible future action of the ego object
- **Play**
  Consider the interaction between prediction and ego motion planning, refine if necessary

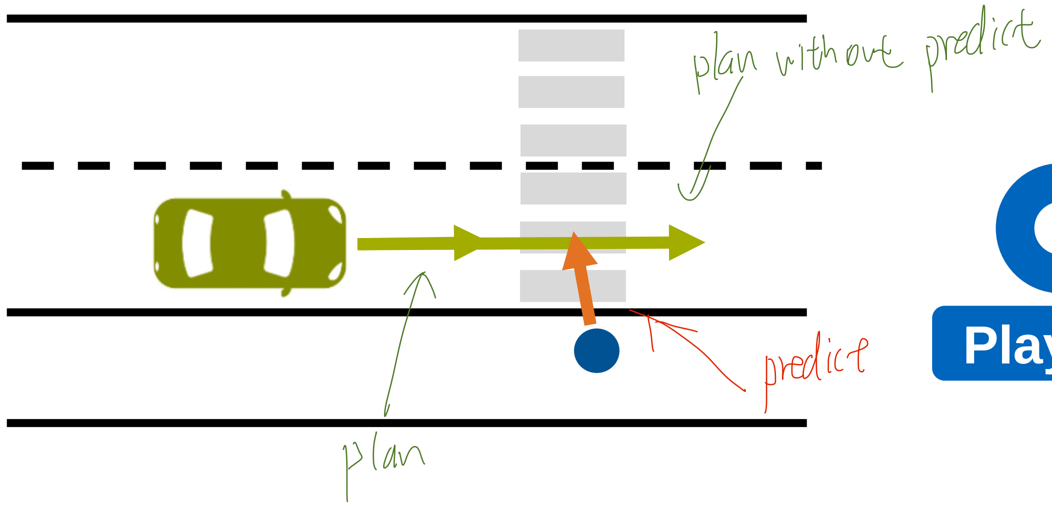


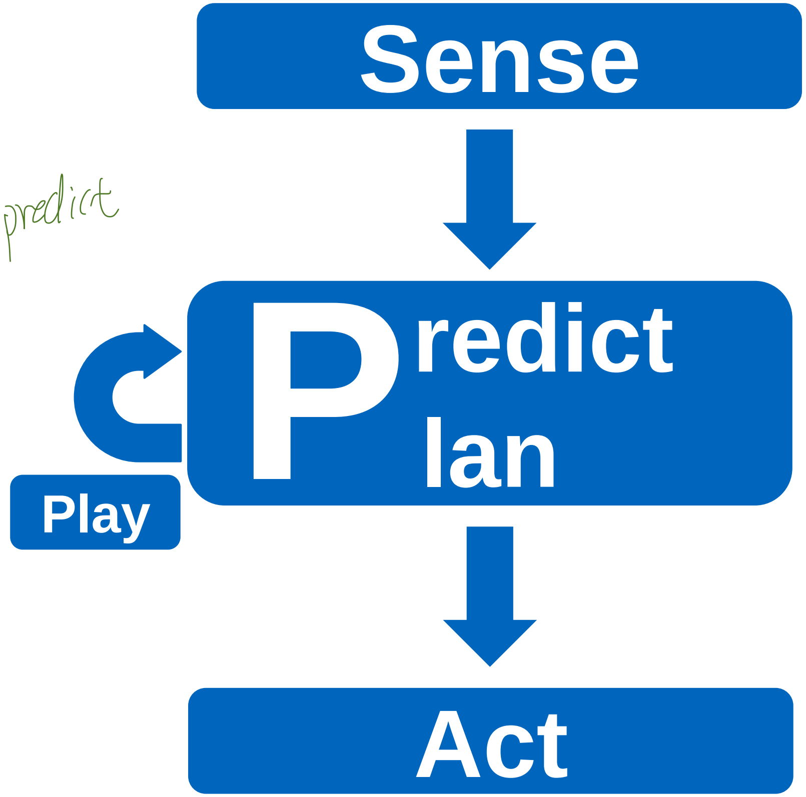
Foundations – Why is prediction needed?





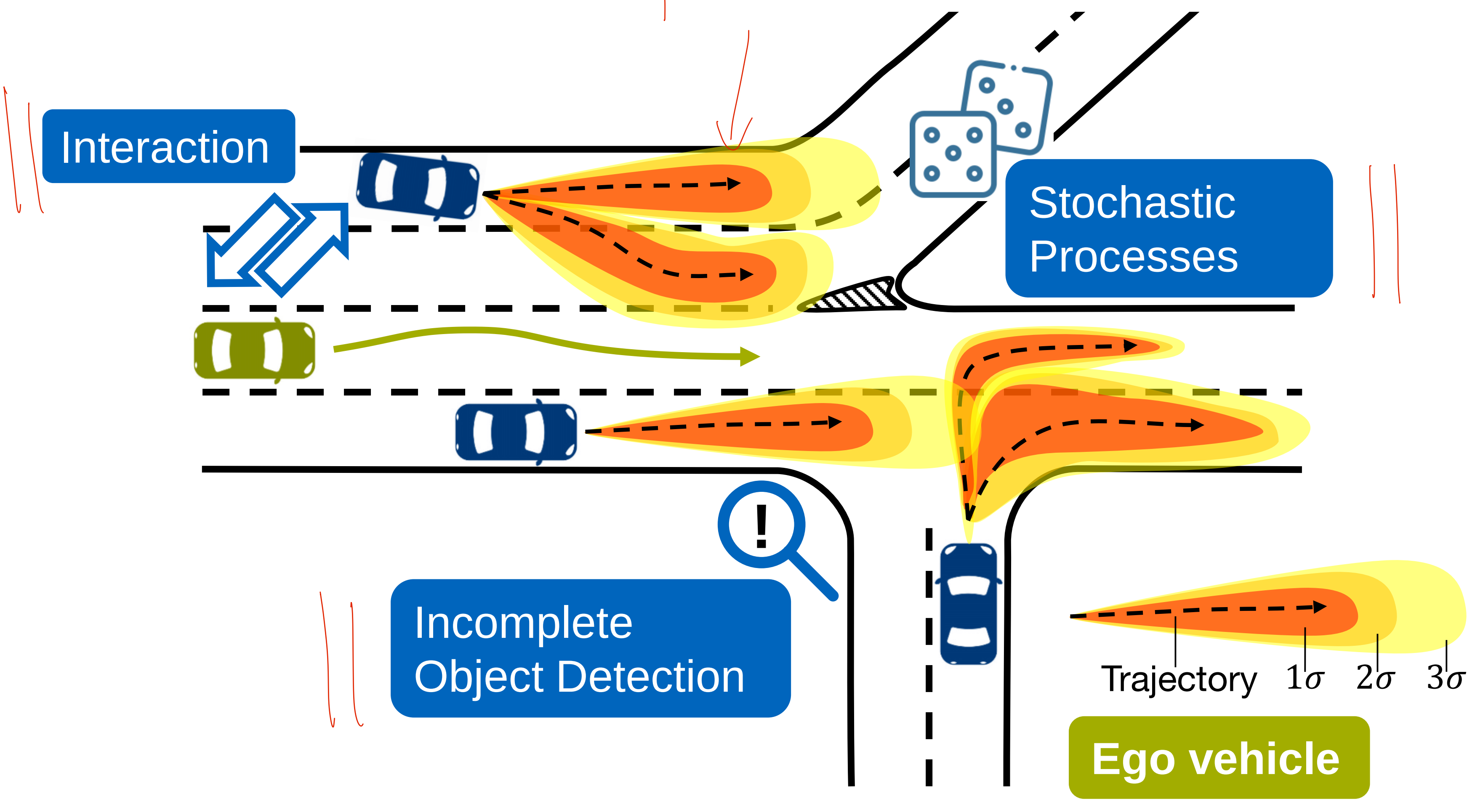
Foundations – Why is prediction needed?





Foundations – Prediction challenges

predict



Foundations – Consequences for ego planning

Risk analysis

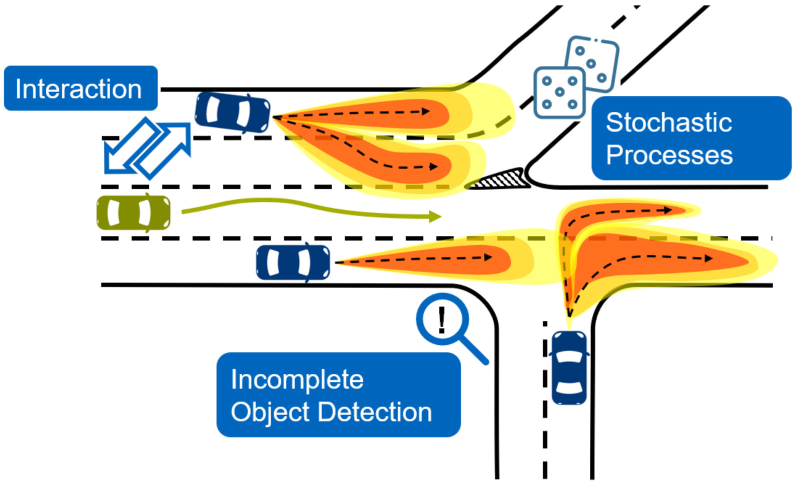
Uncertainty results in risk of planned ego decisions

Collision avoidance

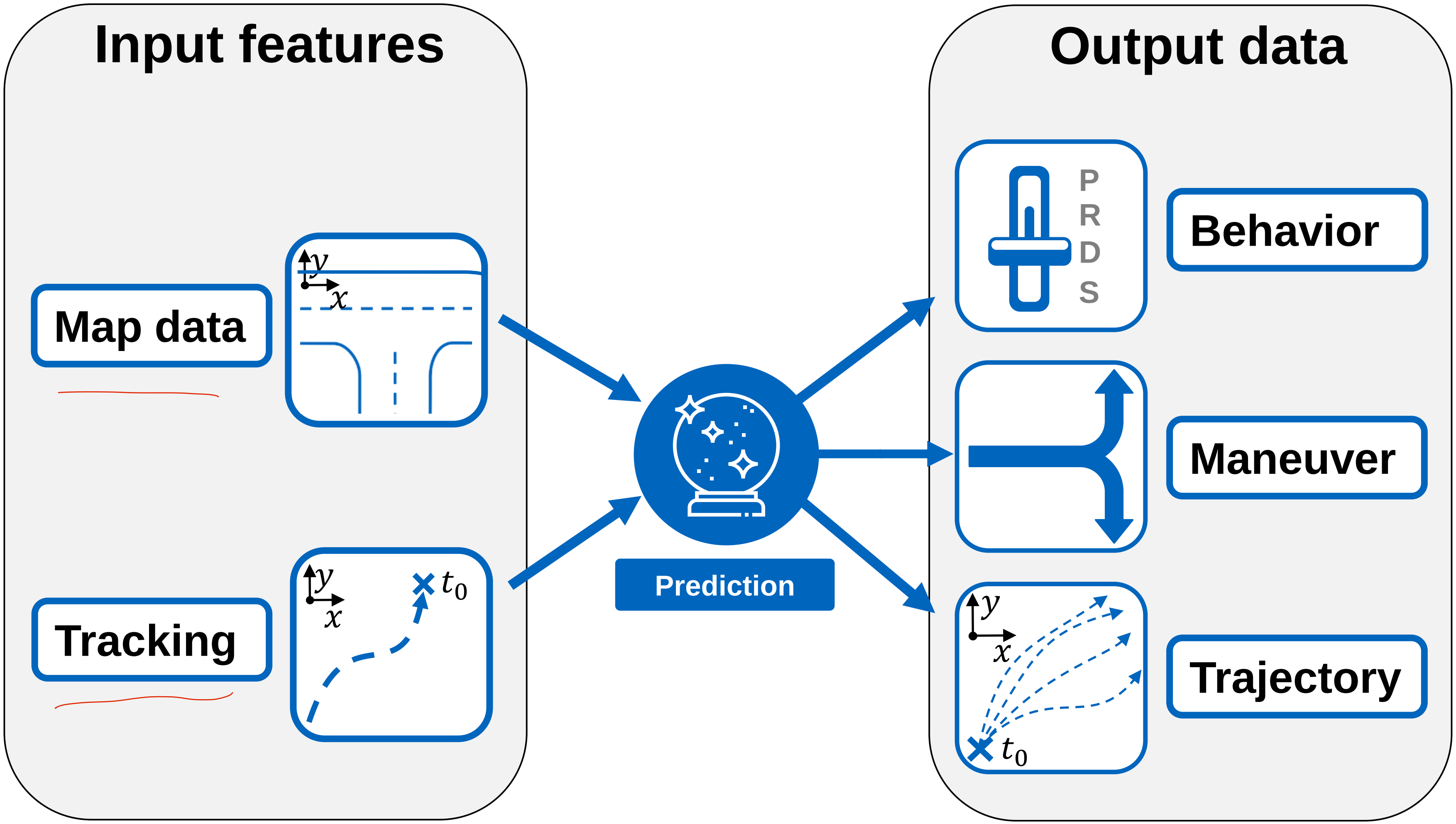
Collision probabilities have to be quantified to ensure safety of traffic participants

Limited degree of freedom

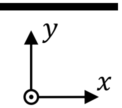
If complexity in current situation is too high, no valid motion is possible
→ Freezing Robot Problem



Interfaces of the Prediction Module



Where is the car going?

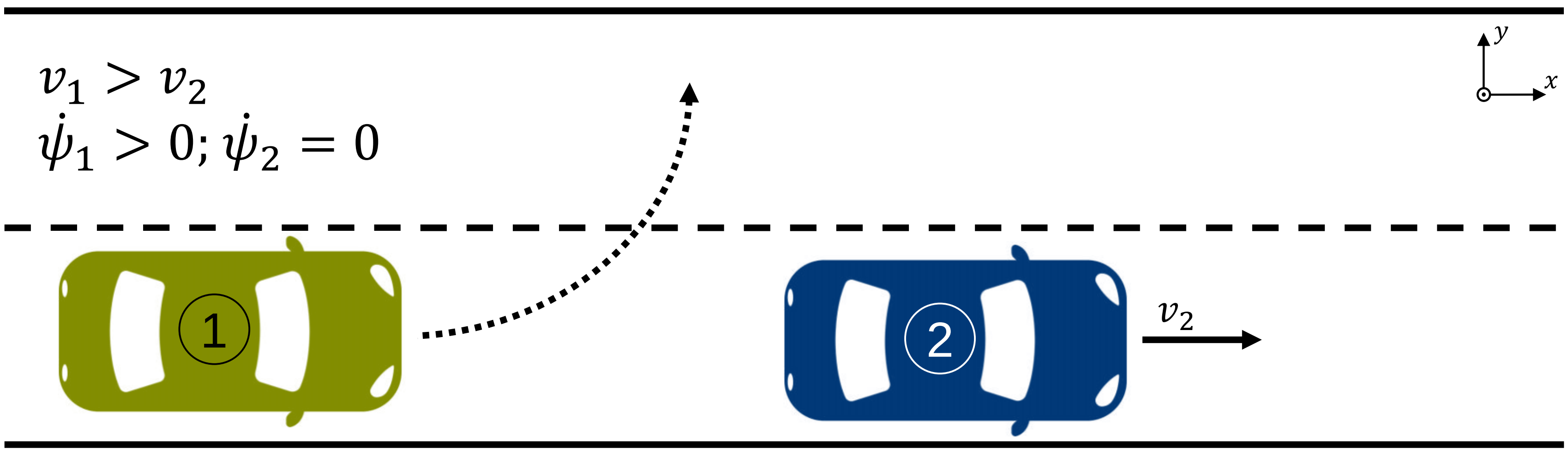


$$v_1 > v_2$$
$$\dot{\psi}_1 > 0; \dot{\psi}_2 = 0$$

Where is the car going? – Knowledge-based

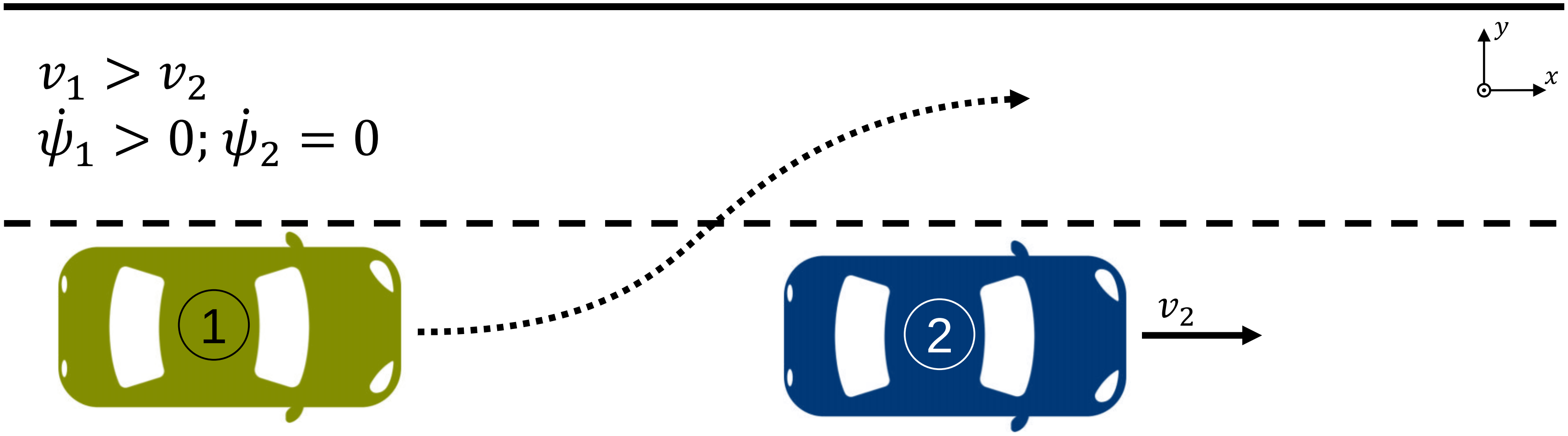
Assumption of constant velocity and yaw rate

$$v_1 > v_2$$
$$\dot{\psi}_1 > 0; \dot{\psi}_2 = 0$$



Where is the car going? – Learning-based

Similar scenarios observed in the dataset results in overtaking prediction



Classes of Prediction Models

Class	Concept	Methods
Knowledge-based Prediction	Sense – Predict	• State Estimation • Reachable Sets
Learning-based Prediction	Sense – Learn – Predict	• Clustering & Classification • Deep Learning • Inverse Reinforcement Learning

Prediction
Prof. Dr. Markus Lienkamp

Dipl.-Ing. Nico Uhlemann

Agenda

1. Foundations
2. **Knowledge-Based Prediction**
   a. **State Estimation**
   b. **Reachable Sets**
3. Learning-Based Prediction
   a. Clustering and Classification
   b. Deep Learning
   c. Inverse Reinforcement Learning
4. Summary and Outlook

a. State Estimation: Introduction

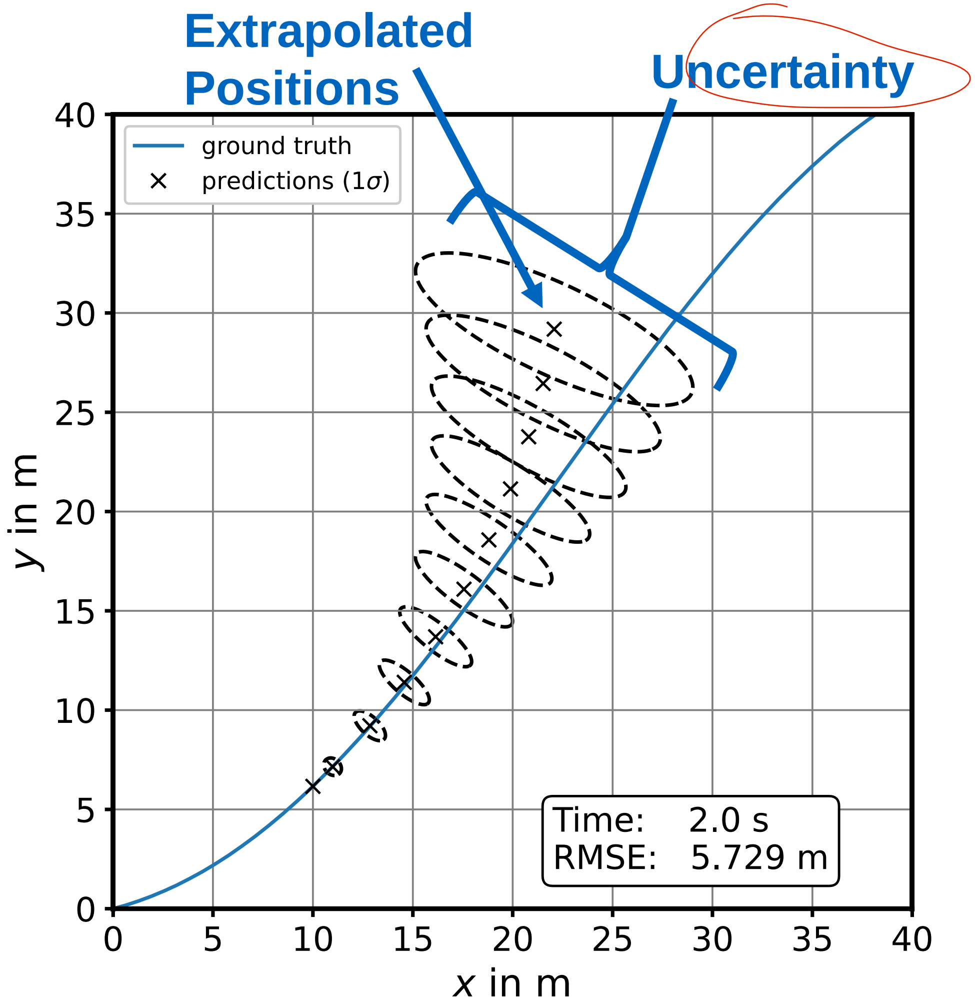
Idea

- Prediction of future positions by physics-based models
- Quantification of uncertainty by means of Bayesian filter

Output: Trajectory Prediction

Application

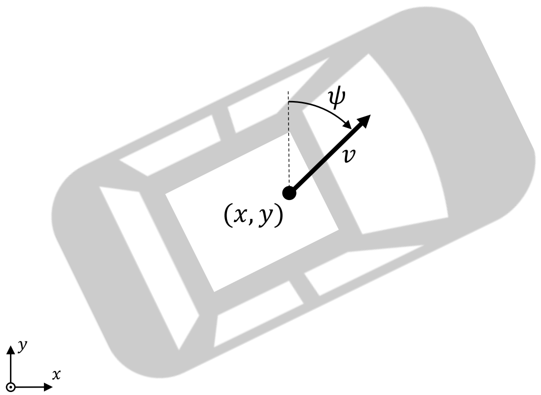
- Short-Term prediction
- Collision check



a. State Estimation: State-Space Model

Discrete State-Space Model

- Initial State $\boldsymbol{x}_{t_0} = \boldsymbol{x}_0$
- State Variables $\boldsymbol{x}$
- Input $\boldsymbol{u}$
- Linear Case: $\boldsymbol{x}_{t+1} = \boldsymbol{A}\boldsymbol{x}_t + \boldsymbol{B}\boldsymbol{u}_t$
- General Case: $\boldsymbol{x}_{t+1} = \boldsymbol{F}(\boldsymbol{x}_t, \boldsymbol{u}_t, \Delta t)$



Approach

- Laws of mechanics with simplifications
- Bayesian filter for uncertainty estimation

a. State Estimation: State-Space Model

Laws of Mechanics with Simplifications

Kinematic Models

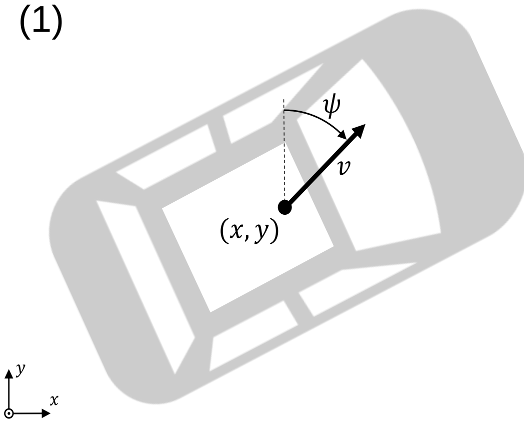
(1) Point-mass model

Dynamic Models

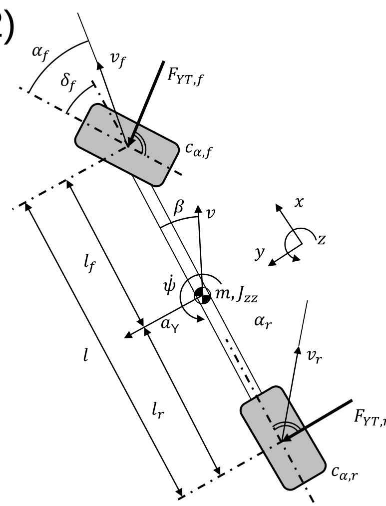
(2) Bicycle Model

(3) Four-Wheel Model

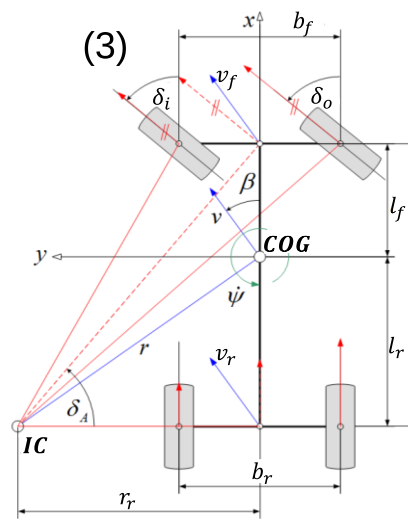
(1)



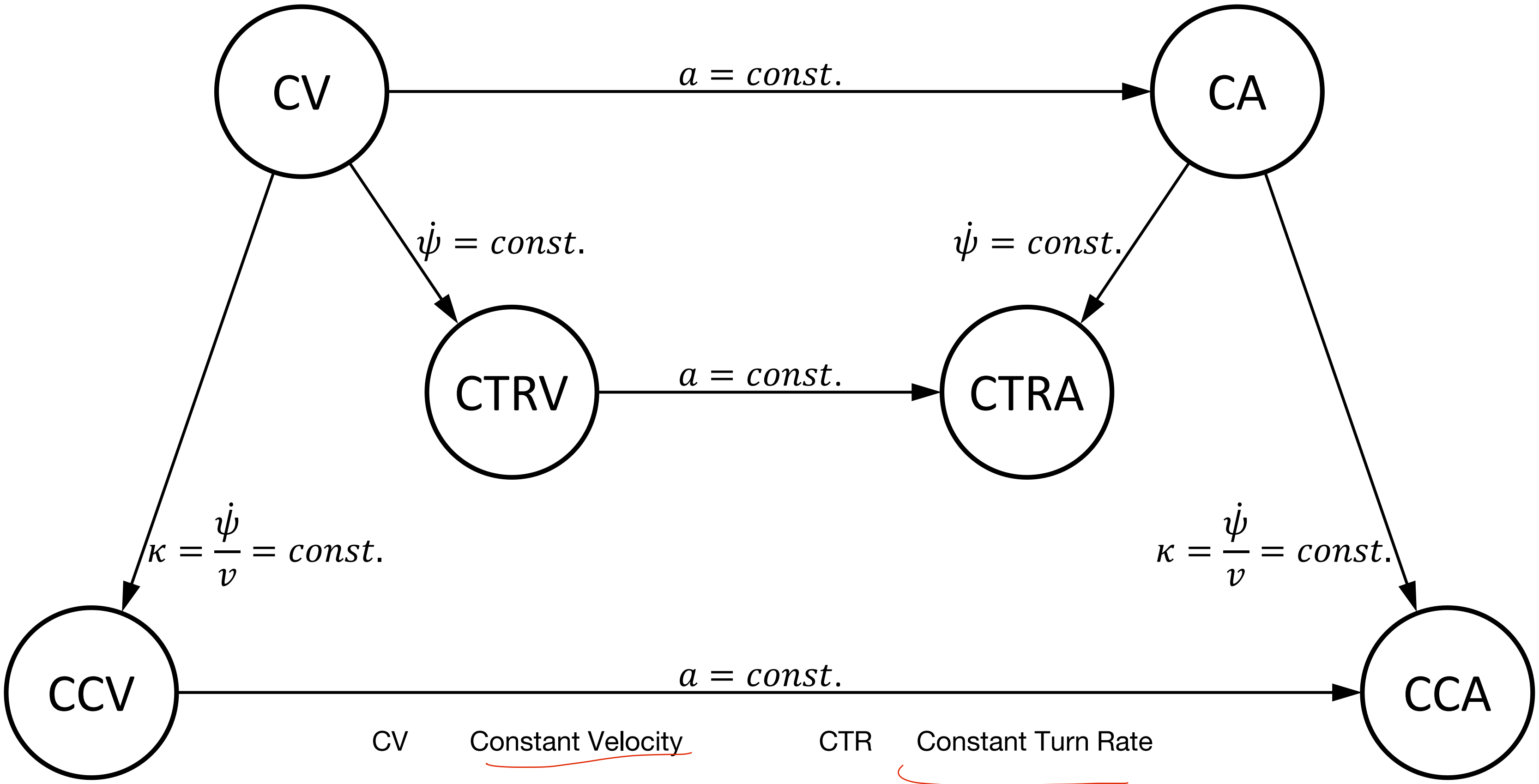
(2)



(3)



a. State Estimation: Basic Kinematic Models

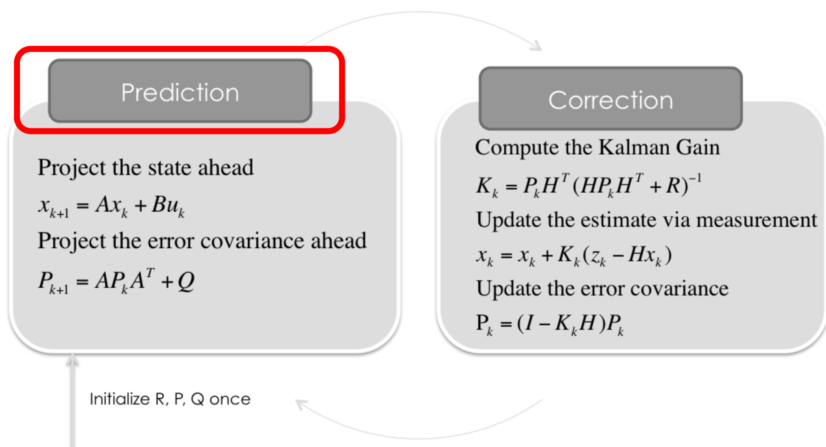


CV	Constant Velocity	CTR	Constant Turn Rate
CC	Constant Curvature	CA	Constant Acceleration
$\dot{\psi}$	Turn Rate	κ	Curvature
a	Acceleration	v	Velocity

a. State Estimation: Bayesian Filter

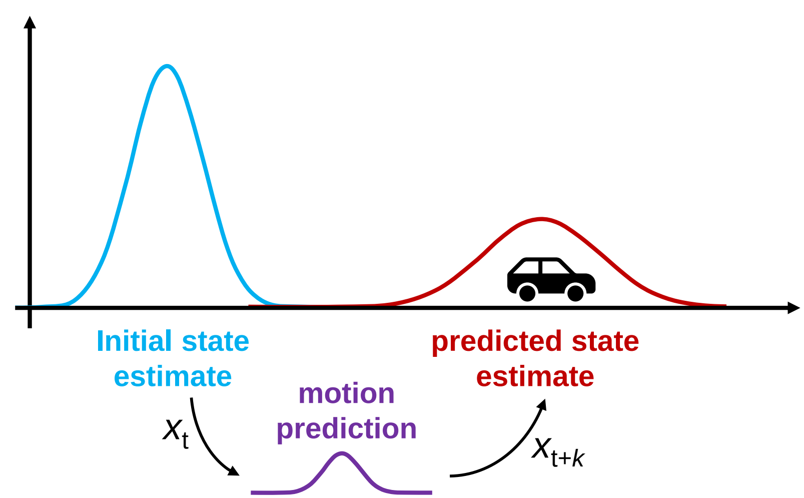
Bayesian Filter for Uncertainty Estimation

- Kalman Filter
- Extended Kalman Filter
- Unscented Kalman Filter
- Particle Filter



For Motion Prediction

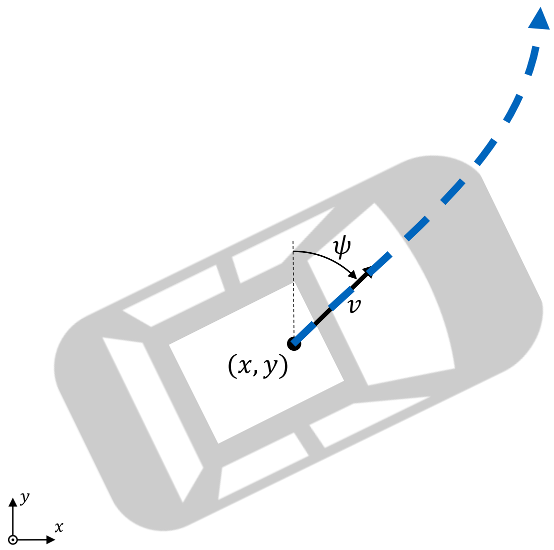
→ Apply prediction step without measurement update



a. State Estimation – Example

State-Space Model: CTRA

$$\begin{pmatrix} x \\ y \\ \Psi \\ v \\ \dot{\Psi} \\ a \end{pmatrix}_{t+k} = \begin{pmatrix} x + \frac{v}{\dot{\Psi}}\left(\cos\left(\dot{\Psi}\Delta t + \Psi\right) - \cos(\Psi)\right) \\ y + \frac{v}{\dot{\Psi}}\left(\sin\left(\dot{\Psi}\Delta t + \Psi\right) - \sin(\Psi)\right) \\ \psi + \dot{\Psi}\Delta t \\ v + a\Delta t \\ \dot{\Psi} \\ a \end{pmatrix}_t$$



State Estimation: Extended Kalman Filter

$$\boldsymbol{x}_{t+1} = \boldsymbol{F}_A(\boldsymbol{x}_t)$$

$$\boldsymbol{P}_{t+1} = \boldsymbol{J}_A \boldsymbol{P}_t \boldsymbol{J}_A^T + \boldsymbol{Q}_t$$

Jacobian Matrix $\boldsymbol{J}_k = \frac{\partial y_i}{\partial x_j}$

Process Noise $\boldsymbol{Q}_t$

State Covariance $\boldsymbol{P}_t$

CTRA: Constant Turn Rate and Acceleration

a. State Estimation – Example

Prediction Horizon

- Time Steps $\{0, 0.2\mathrm{s}, \dots, t_{\mathrm{pred}} = 2.0\mathrm{s}\}$
- Get $\boldsymbol{x}_{\mathrm{pred}}$, $\boldsymbol{P}_{\mathrm{pred}}$

$$\boldsymbol{x}_{\mathrm{pred}} = \begin{pmatrix} x_{\mathrm{p}} \\ y_{\mathrm{p}} \end{pmatrix} = \begin{pmatrix} x_{t_1} \dots x_{t10} \\ y_{t_1} \dots y_{t10} \end{pmatrix}$$

$$\boldsymbol{P}_{\mathrm{pred}} = (\boldsymbol{P}_{t_1} \quad \dots \quad \boldsymbol{P}_{t_{10}})$$

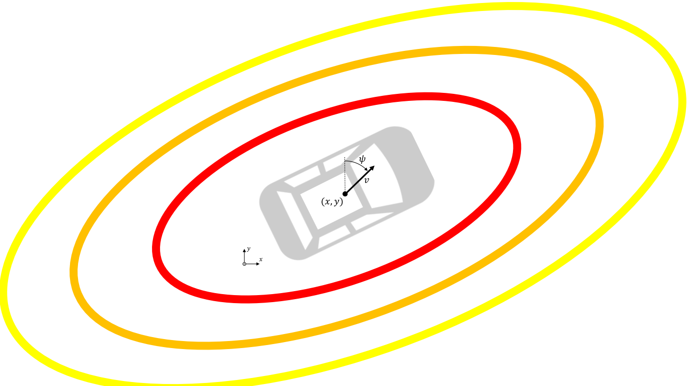
Evaluate Collision Probability

- Uncertainty weighted distance: Mahalanobis Distance
- χ^2-Distribution determines collision probability
- Collision probability has to stay below safety threshold D_{crit}

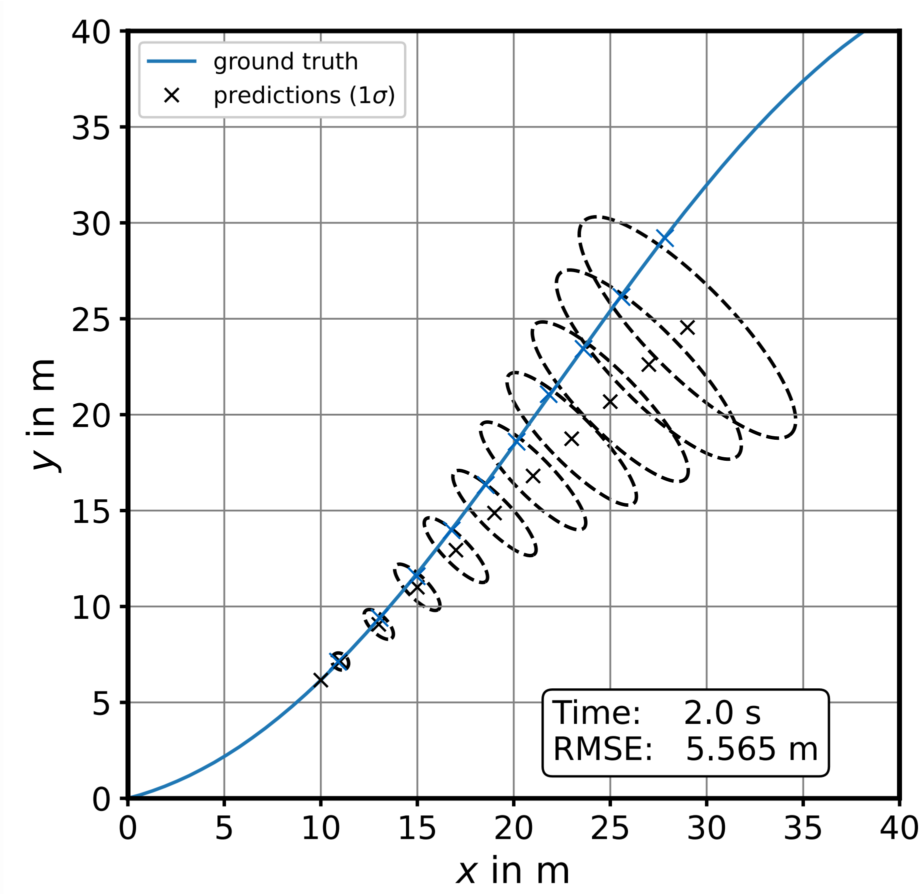
→ Ellipsoid safety region

$$\boxed{D_{\mathrm{MH}}} = \sqrt{\left(\boldsymbol{x}_{t_i} - \boldsymbol{x}_{\mathrm{ego}}\right)^{\mathrm{T}} \boldsymbol{P}^{-1} \left(\boldsymbol{x}_{t_i} - \boldsymbol{x}_{\mathrm{ego}}\right)}$$

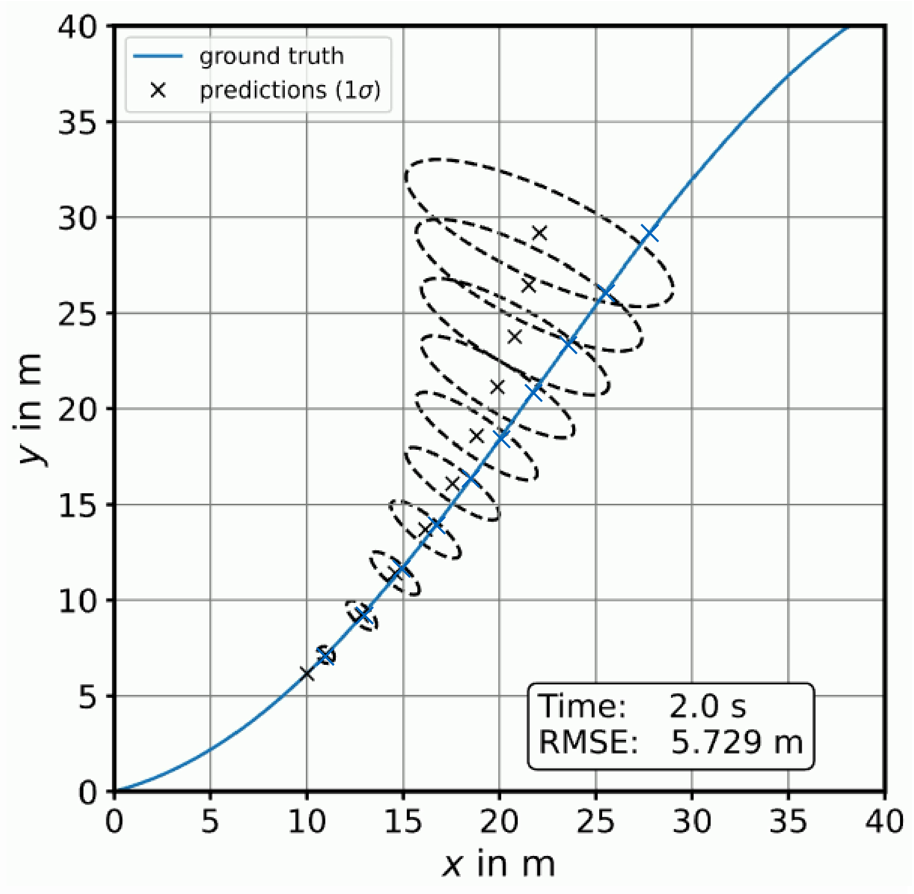
$$D < D_{\mathrm{crit}} = \chi^2$$



a. State Estimation – Comparison



CV-Model



CTRA-Model

CV: Constant Velocity
CTRA: Constant Turn Rate and Acceleration
RMSE: Root Mean Square Error

b. Reachability Analysis: Introduction

Idea

- Derivation of all possible positions within physical constraints
- Over approximation of reachable positions
- Application of traffic rules to shrink set size

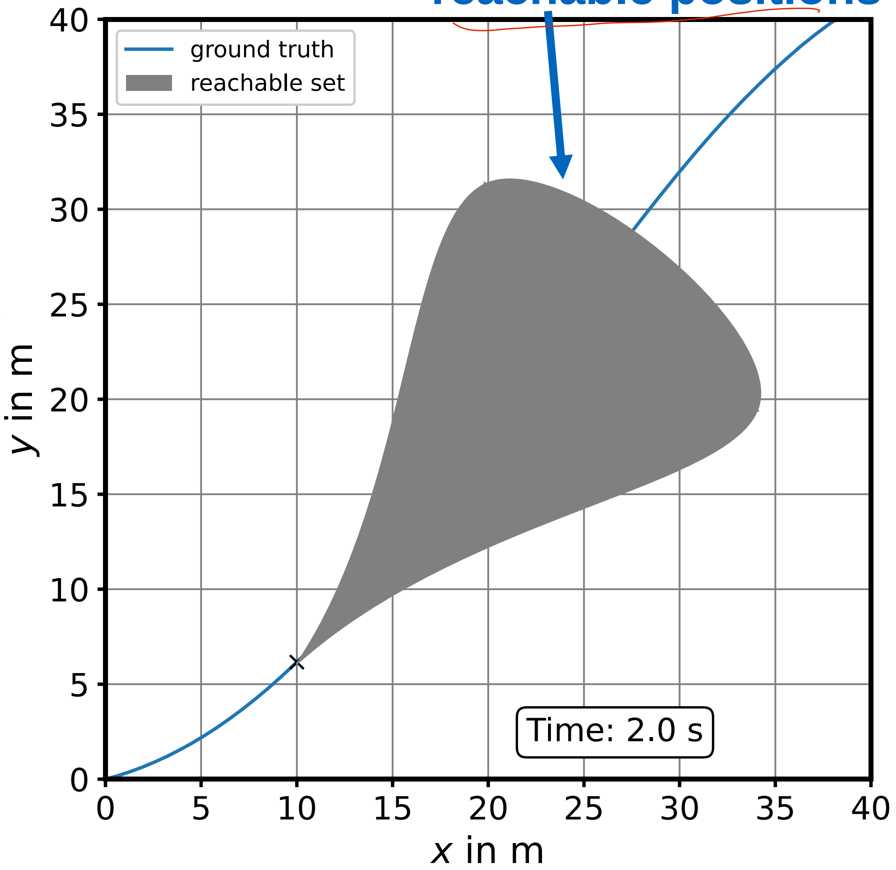
Output

- Occupancy Map

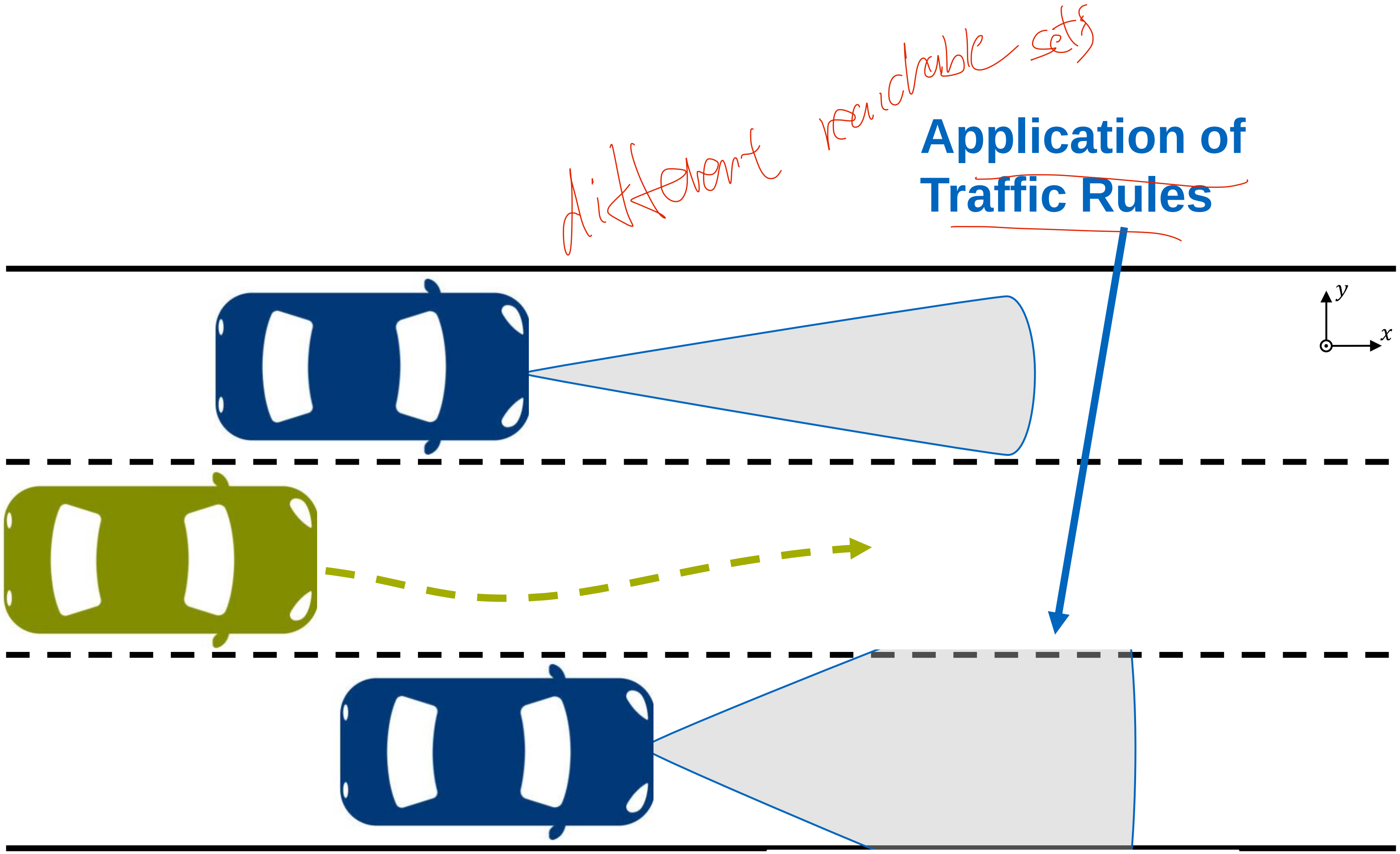
Application

- Online Verification
- Safety Assessment

Set of all reachable positions



b. Reachable Sets



Prediction
Prof. Dr. Markus Lienkamp

Dipl.-Ing. Nico Uhlemann

Agenda

1. Foundations
2. Knowledge-Based Prediction
   a. State Estimation
   b. Reachable Sets
3. **Learning-Based Prediction**
   a. Clustering and Classification
   b. Deep Learning
   c. Inverse Reinforcement Learning
4. Summary and Outlook

a. Pattern Clustering

Idea

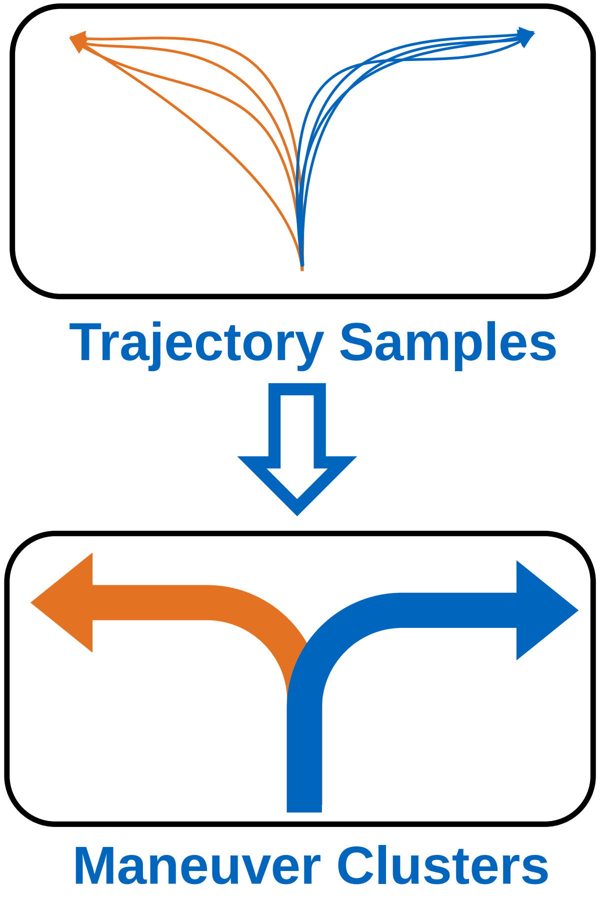
- Clustering of trajectories into few data classes
- Cluster data into maneuvers

Output

- Maneuver Cluster
- Prototype Trajectory

Application

- Input for classification-based prediction methods
- Dimension Reduction

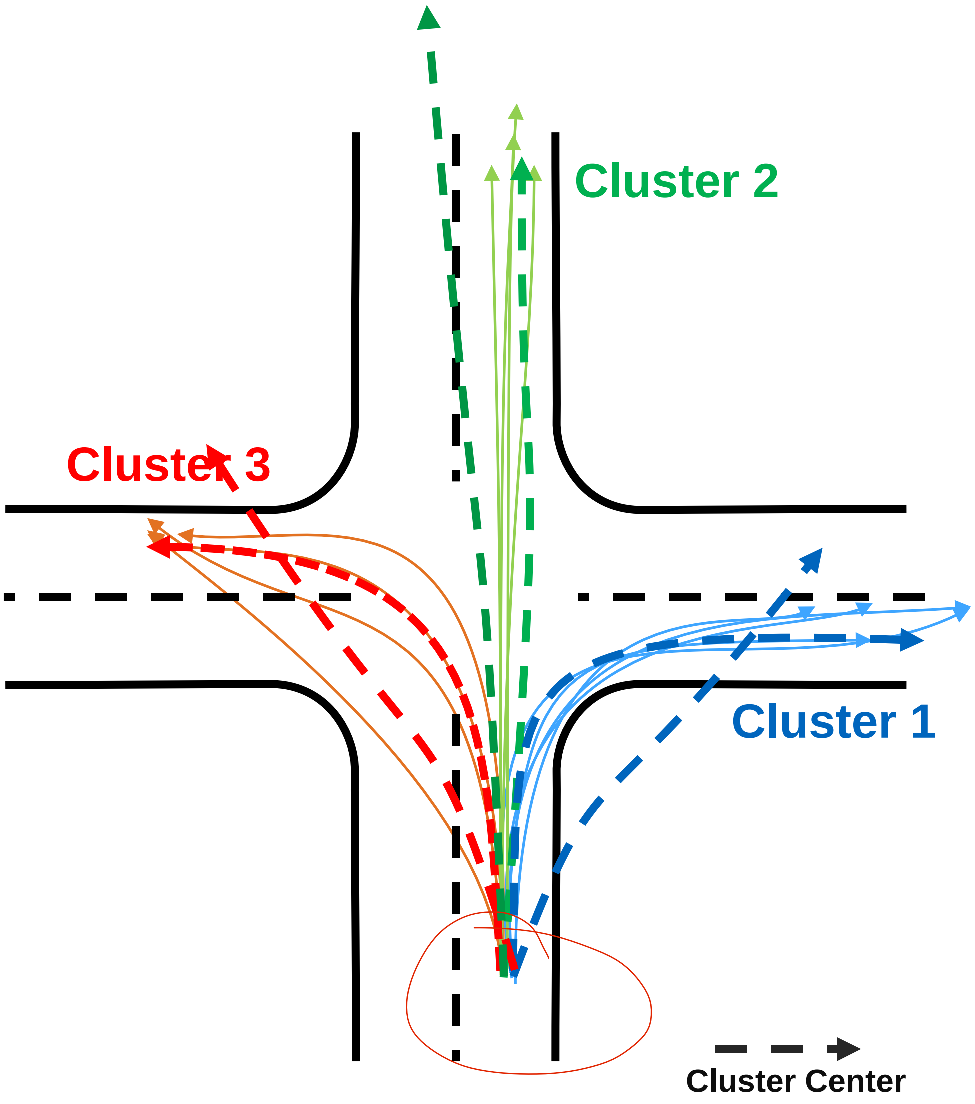


a. Pattern Clustering

K-means
Top-Down Clustering

Algorithm

- k initial random points
- Associate each sample to a center point
- Update new center point as mean of associated points
- Iterate until convergence

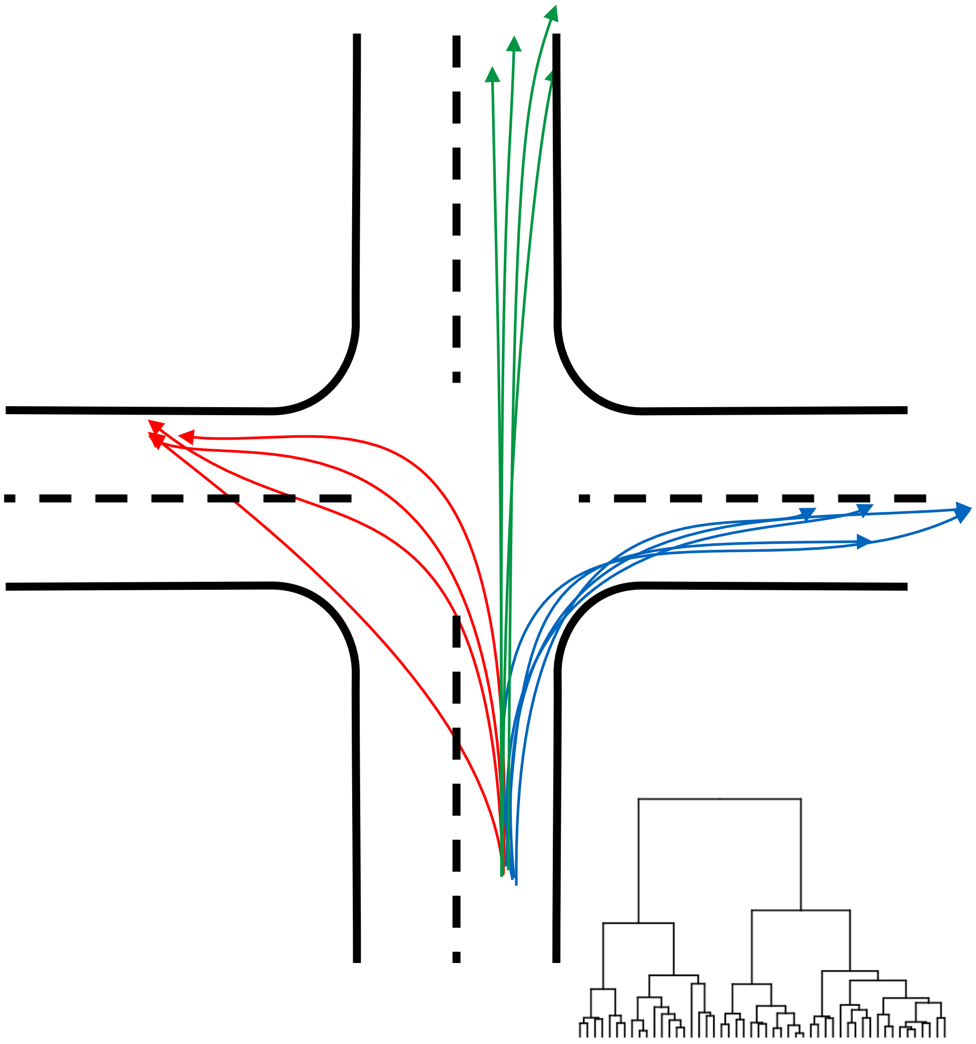


a. Pattern Clustering

Agglomerative Clustering: Bottom-Up

Algorithm

- Each point is one cluster
- Two nearest clusters are combined with new center point
- Distance between clusters based on different metrics (mean, max, min, etc.)
- Iterate until the distance between each cluster surpass the predefined distance



a. Pattern Classification

label

Idea

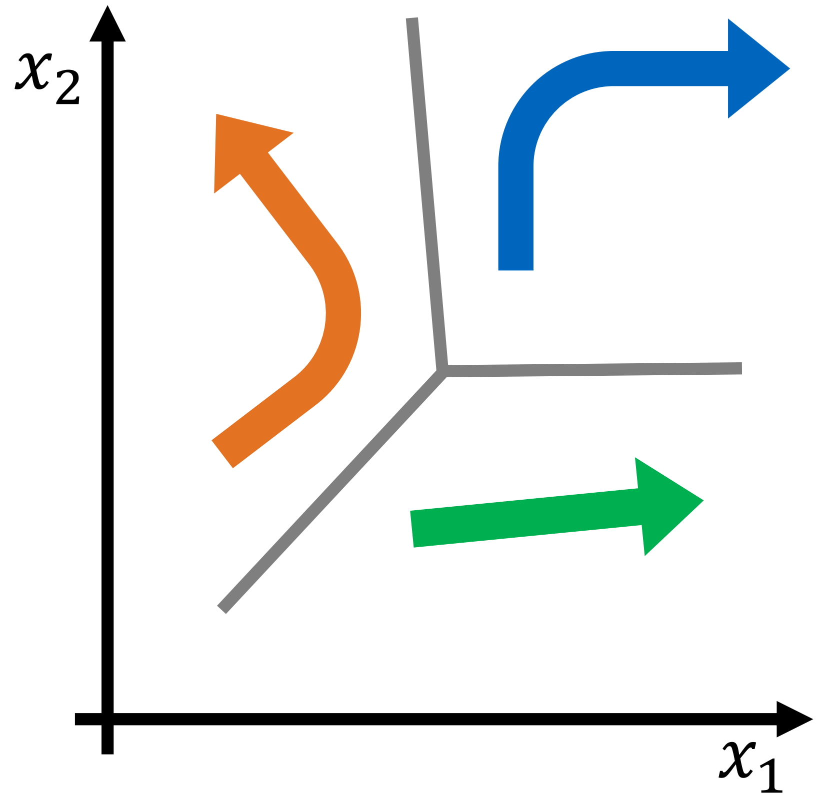
- Classification of new received trajectory into pre-defined classes
- A priori knowledge of outcome: Labeled data necessary for training

Output

- Discrete maneuver classification

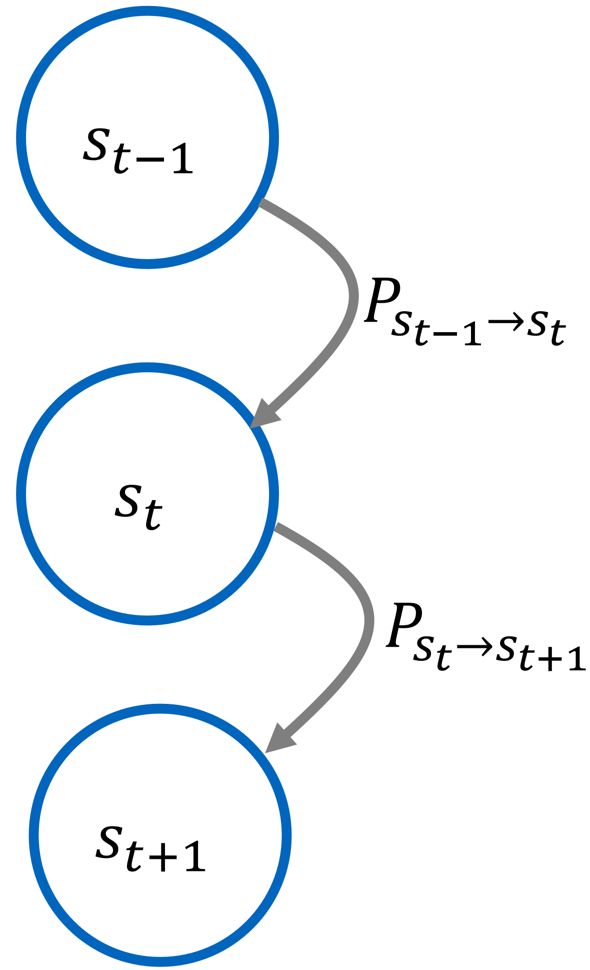
Application

- Maneuver Prediction
  - Lane Change
  - Intention at intersections
- Interactive Planning (Game Theory)



x_i: features

a. Pattern Classification - Markov Models



Idea

- Stochastic model of discrete processes
- Markov Assumption: memoryless property of a stochastic process

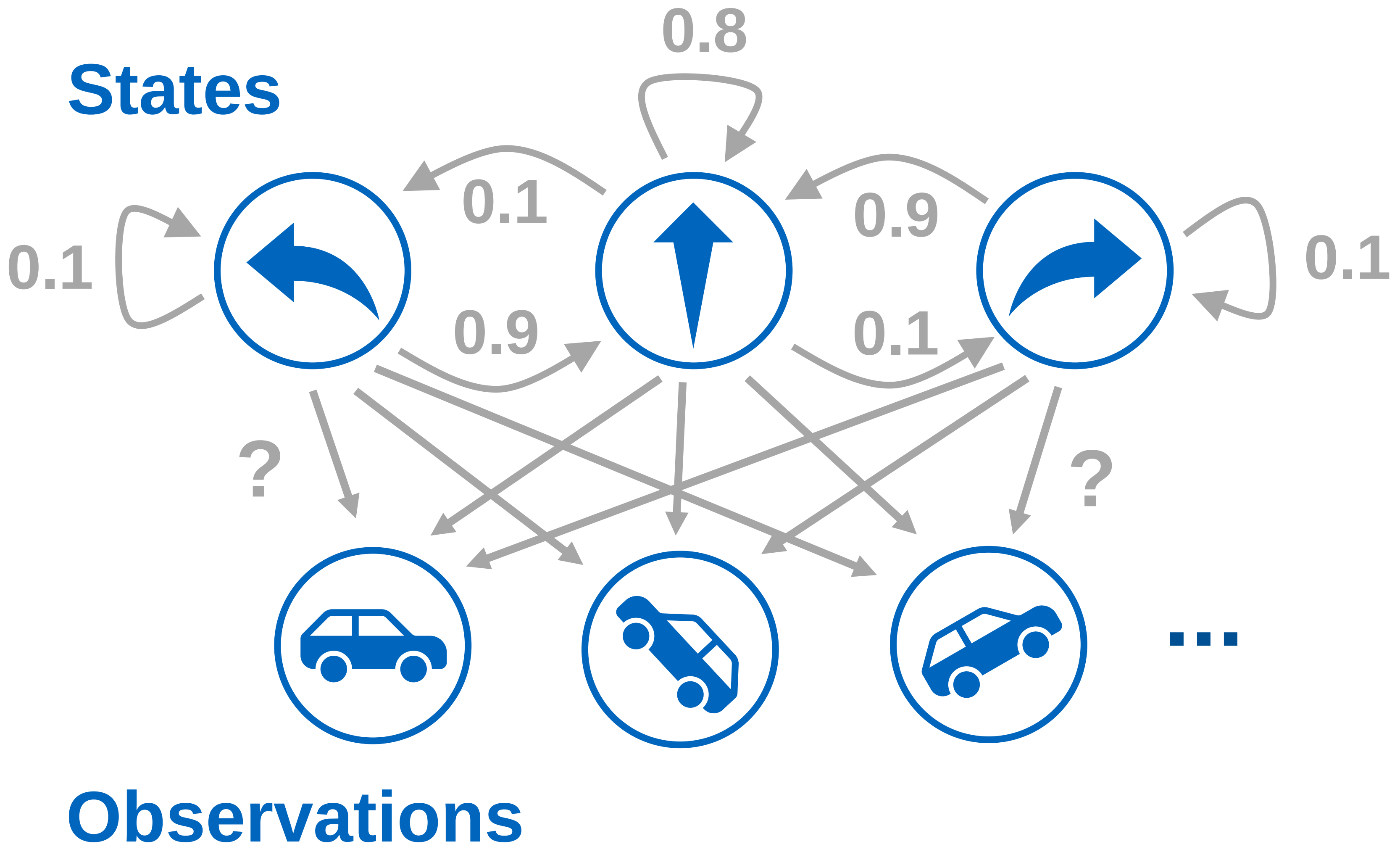
Definition

A stochastic process is a Markov process if the conditional probability distribution of future states of the process depends only upon the present state, not on the sequence of events that preceded it.

$$P\left(s_{t+1} | s_{t,t-1:1}\right) = P(s_{t+1} | s_t);\ \forall t$$

Transition Probability *P*
State *s*

a. Pattern Classification - Markov Models Example



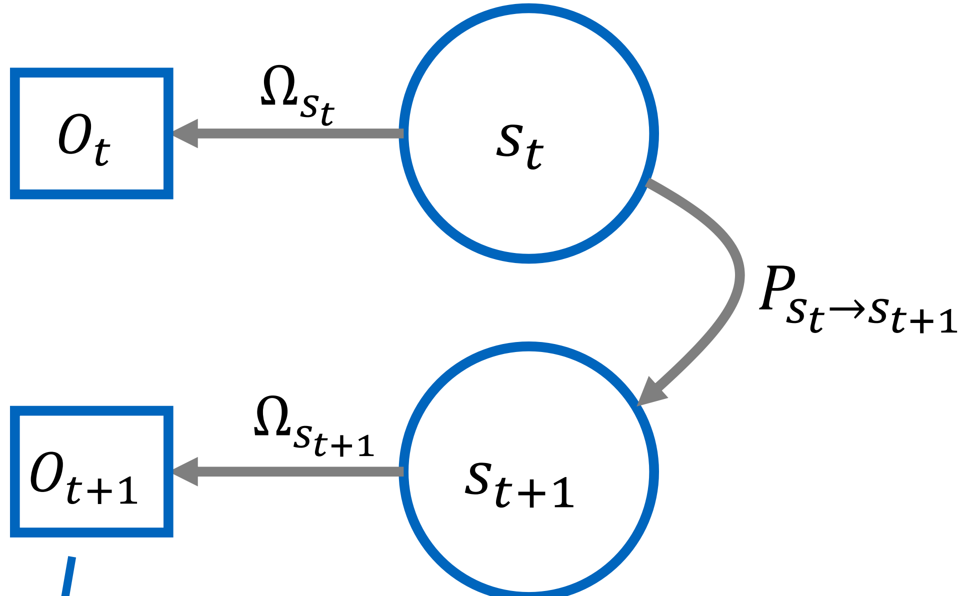
a. Pattern Classification - Hidden Markov Model

Definition “Hidden”

- States of the Markov process are unknown
- Only observations available

Model Parameters

- Tuple: (S, P, Ω, O)
- States $S = \{s_0, \dots, s_n\}$
- Transition probability $P(s'|s)$
- Observation $O(t)$
- Observation model $\Omega(s)$



Measurement
$O_t: x_t, y_t, v_t, \Psi_t, a_t$

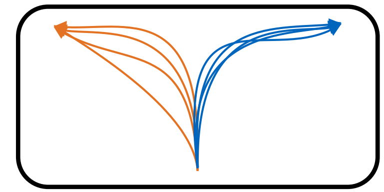
	Predicted direction		
Direction driven	left	right	straight
left	71	1	1
right	11	197	3
straight	1	11	537
mean prediction time	8.3s	4.7s	8.3s

T. Streubel and K. H. Hoffmann, “Prediction of driver intended path at intersections,” in *IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 134–139.

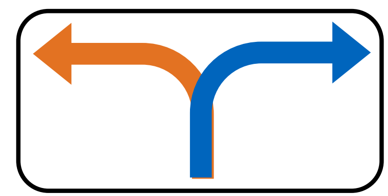
a. Clustering and Classification

+ Discrete, low-dimensional solution space
+ Applicable for structured environments, e.g. highway
+ Applicable for interactive planning (game theory)

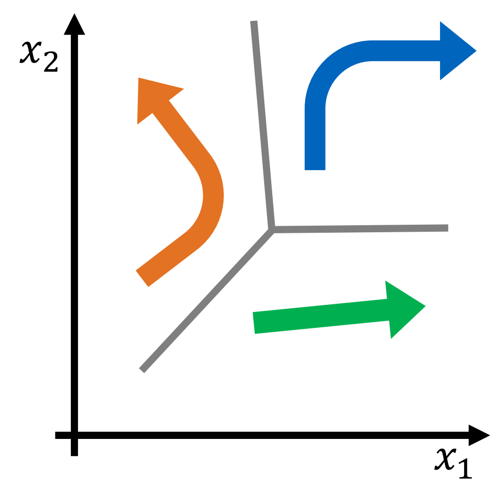
- Clustering: Number and shape of classes unknown
- Classification: high dependency on a priori class definition
- No trajectory prediction possible
- Complexity of road traffic is hard to cover by few classes



Trajectory Samples



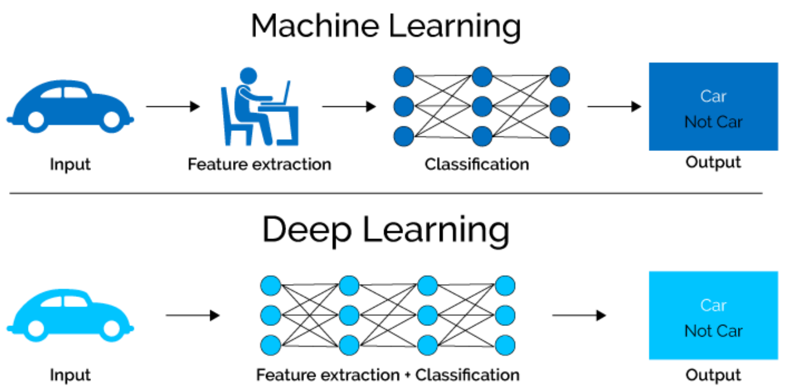
Maneuver Clusters



b. Deep Learning – Motivation

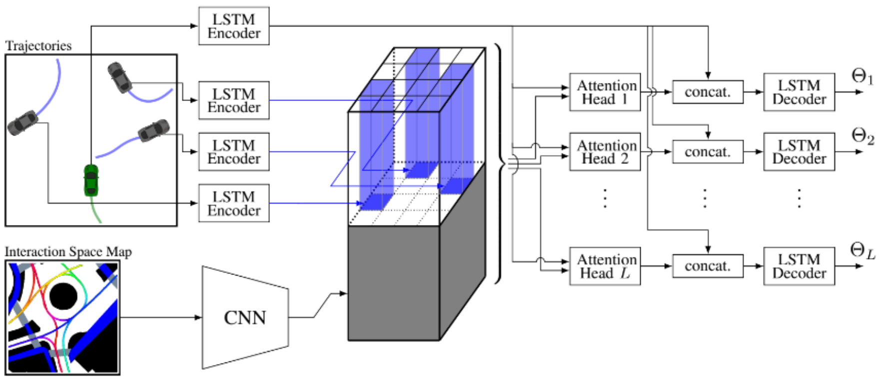
Why Deep Learning?

- Maximum utilization of data
- Creation of new features
  → But: Reason about valid input



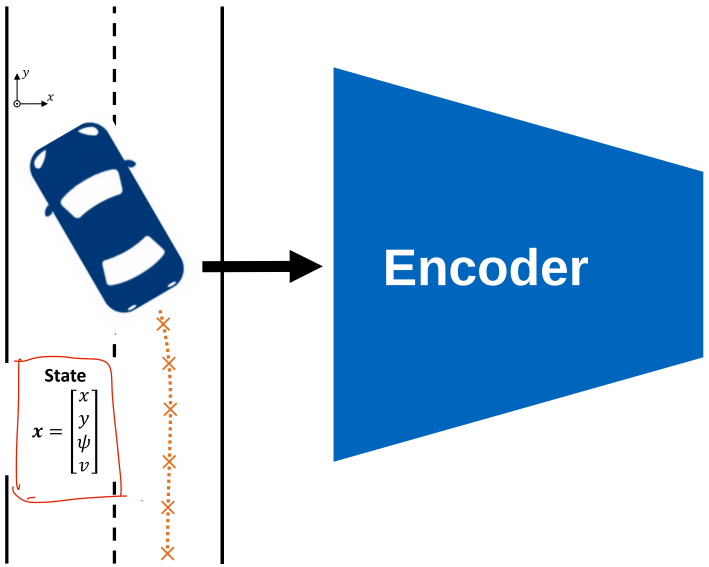
Which Architectures?

- **Encoder-Decoder** (dominated ICRA 2020 prediction challenge)
- **Transformer** (recent success in sequence-to-sequence modelling tasks)
- **Graph Neural Networks** (good performance in modelling interactions)

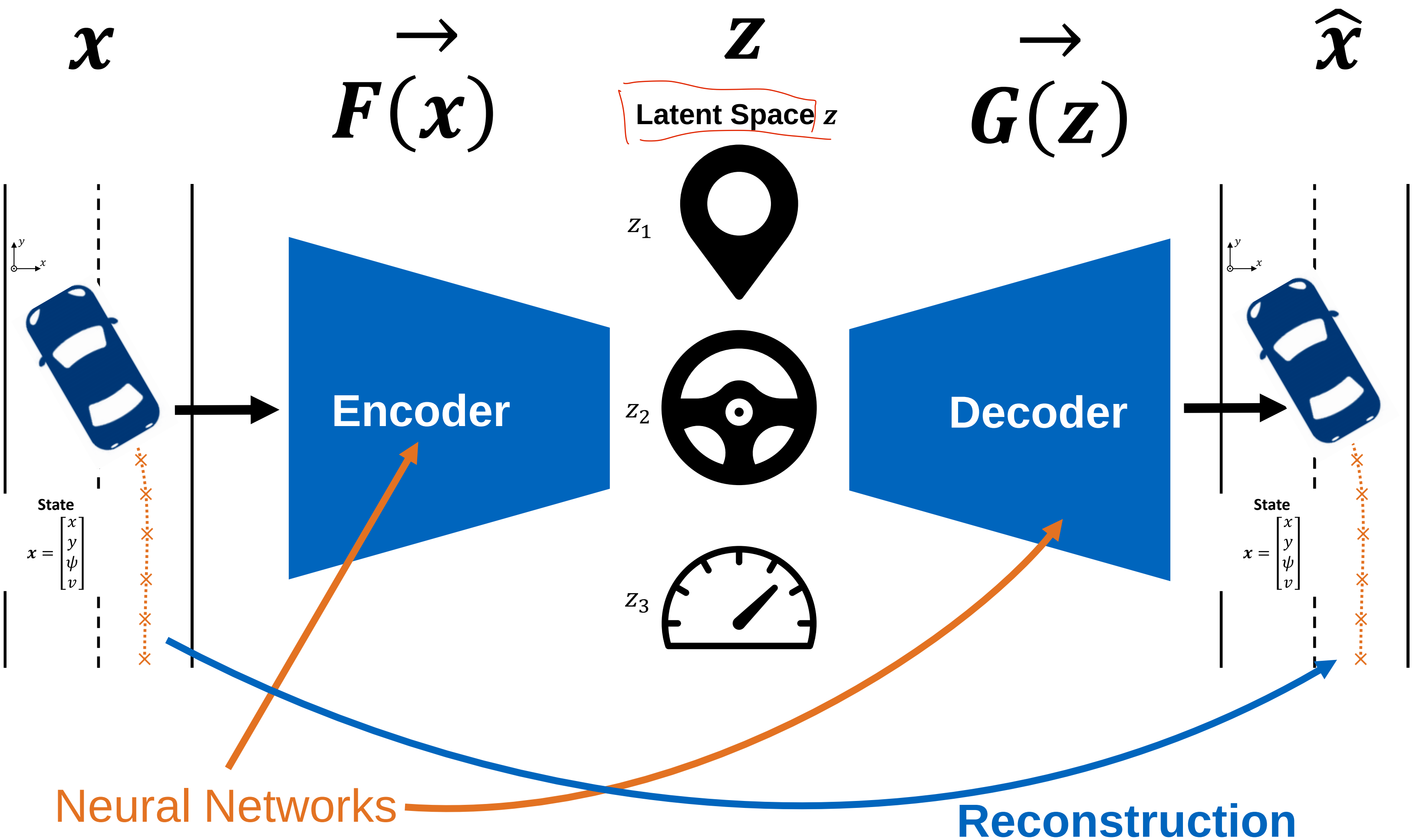


b. Deep Learning – Encoder-Decoder

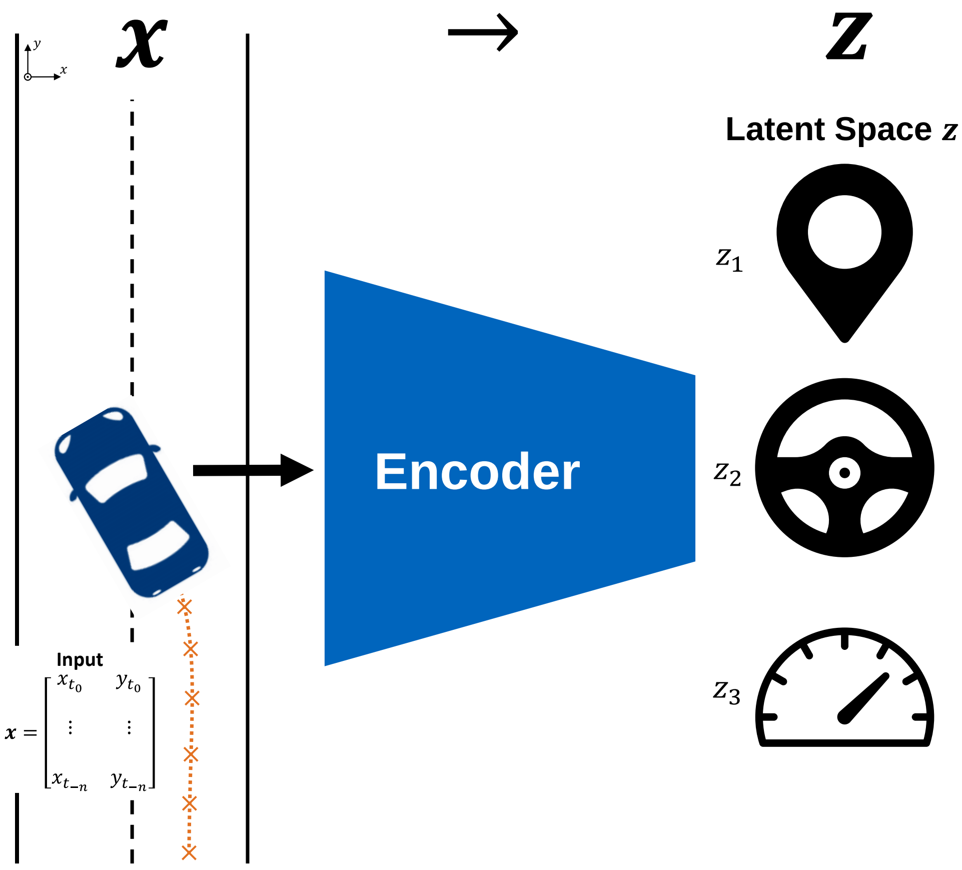
$\boldsymbol{x} \quad \rightarrow$



b. Deep Learning – Encoder-Decoder

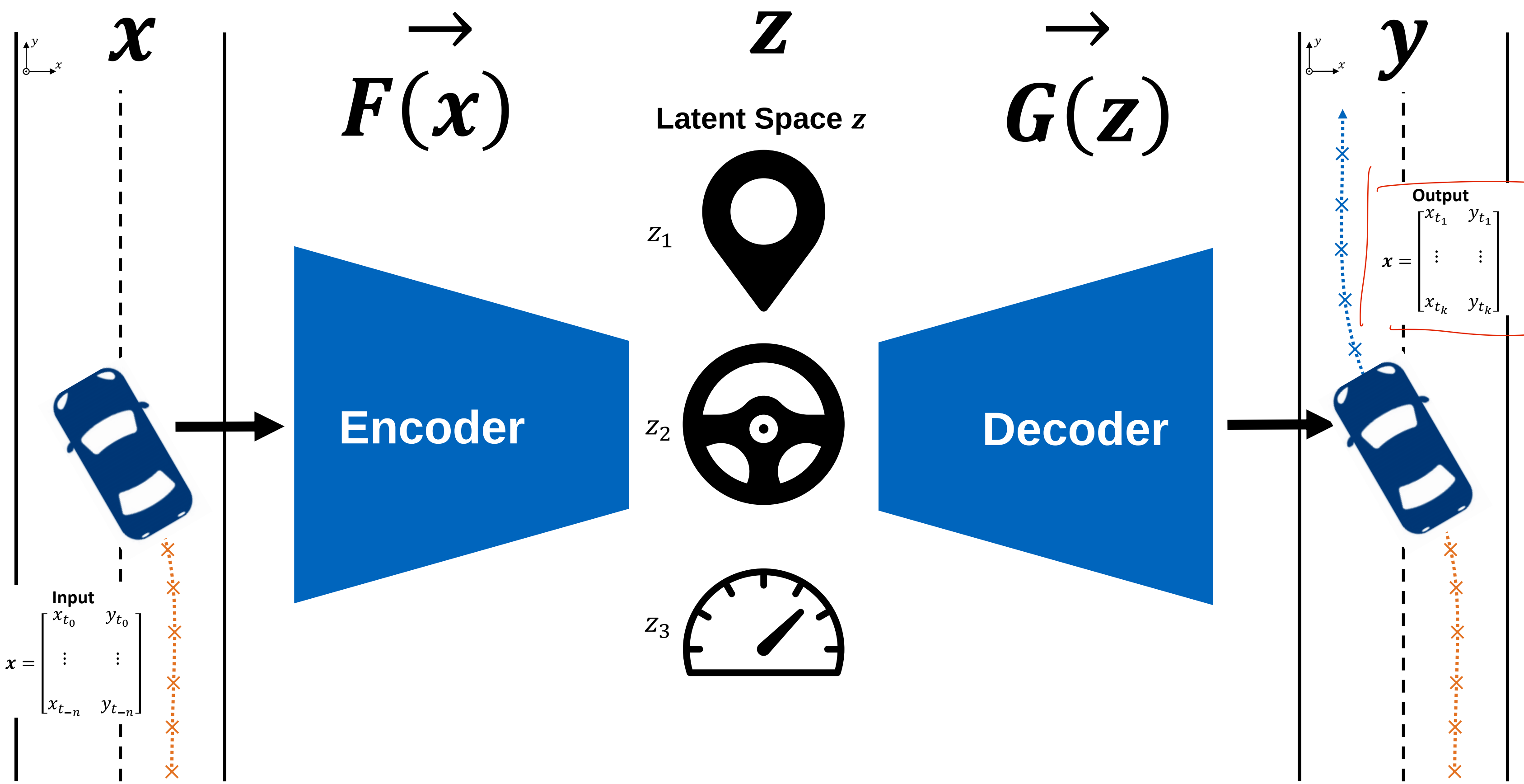


b. Deep Learning – Encoder-Decoder



b. Deep Learning – Encoder-Decoder

predice features



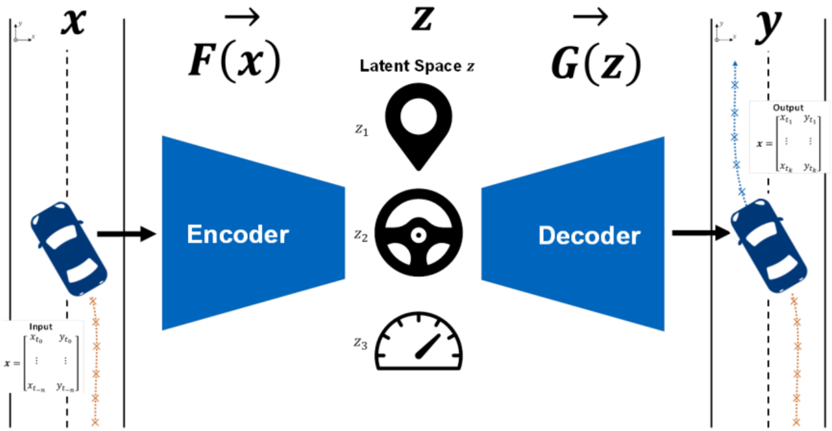
b. Deep Learning – Encoder-Decoder

Loss-Function (e. g. $\|.\|_2^2$-Norm):

$$\mathcal{L}_2(\boldsymbol{y}_{\mathrm{P}}, \boldsymbol{y}_{\mathrm{GT}}) = \|\boldsymbol{y}_{\mathrm{P}} - \boldsymbol{y}_{\mathrm{GT}}\|_2^2$$

with

$$\boldsymbol{y}_{\mathrm{P}} = \begin{bmatrix} x_{t_1} & y_{t_1} \\ \vdots & \vdots \\ x_{t_k} & y_{t_k} \end{bmatrix}_{\mathrm{P}}; \quad \boldsymbol{y}_{\mathrm{GT}} = \begin{bmatrix} x_{t_1} & y_{t_1} \\ \vdots & \vdots \\ x_{t_k} & y_{t_k} \end{bmatrix}_{\mathrm{GT}}$$



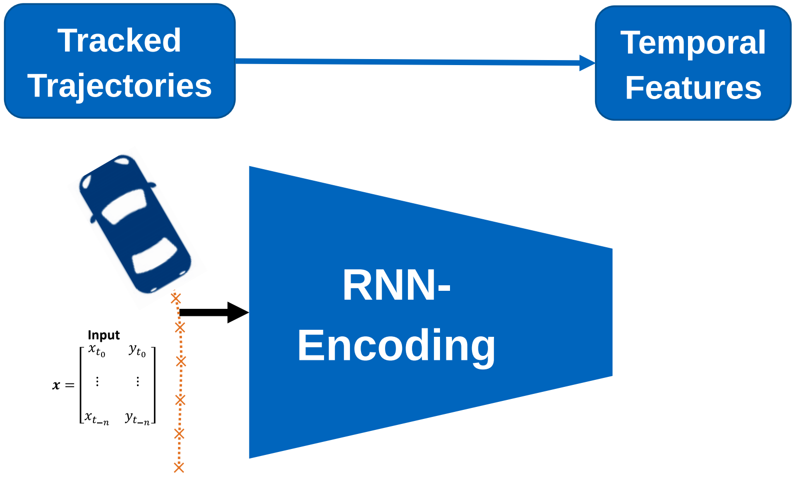
Other applications

- Dimensional Reduction
- Features Extraction
- Image Denoising

$\boldsymbol{x}$:	Input	$\cdot_{\boldsymbol{p}}$:	prediction
$\boldsymbol{y}$:	Prediction	$\cdot_{\boldsymbol{GT}}$:	ground truth

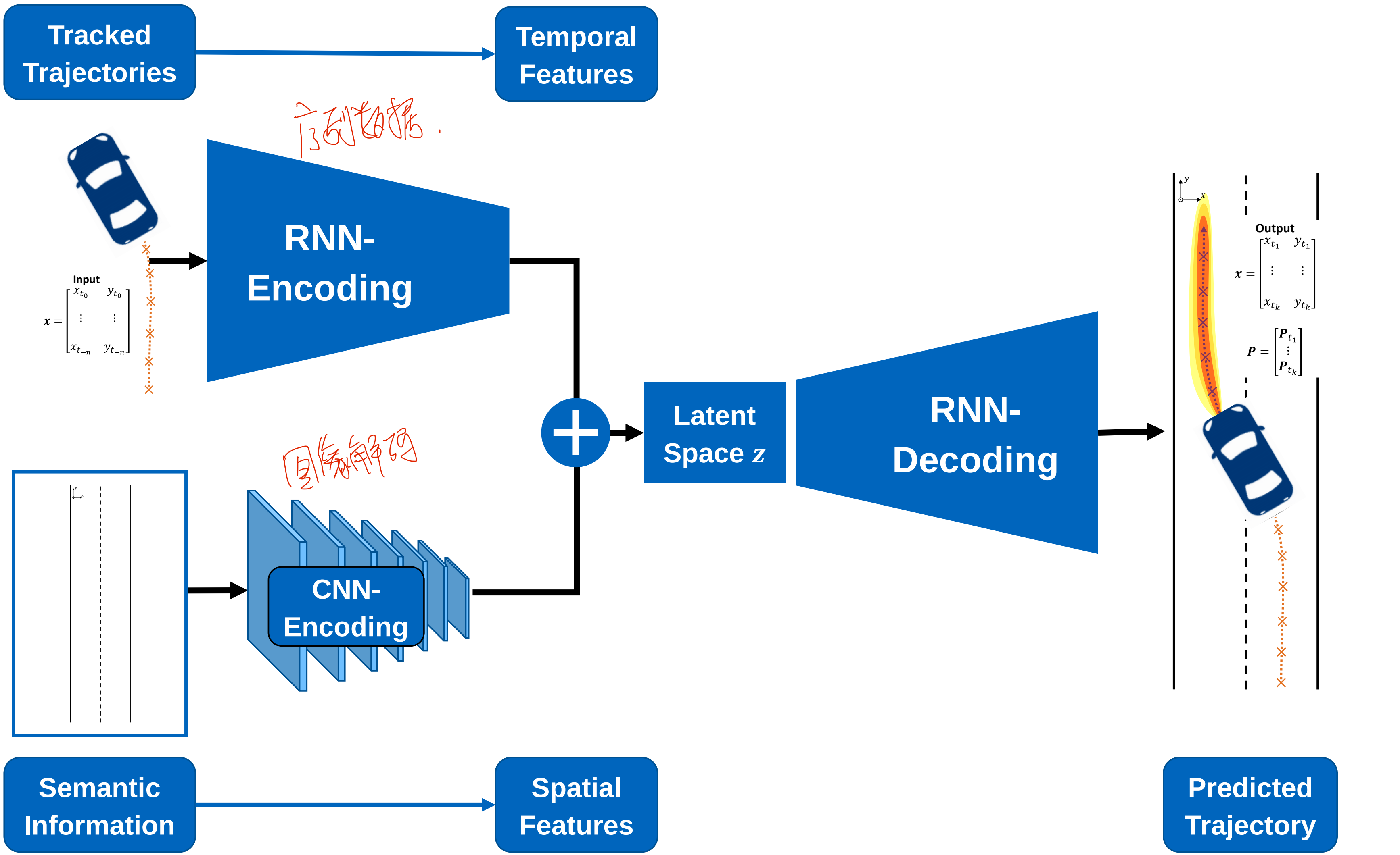
$\boldsymbol{F}(\boldsymbol{x})$:	Encode-Function
$\boldsymbol{z}$:	Latent Space
$\boldsymbol{G}(\boldsymbol{z})$:	Decode-Function
$\mathcal{L}(\boldsymbol{y}_{\mathrm{P}}, \boldsymbol{y}_{\mathrm{GT}})$:	Loss-Function

b. Deep Learning – Encoder-Decoder Example



RNN Recurrent Neural Network
CNN Convolutional Neural Network

b. Deep Learning – Encoder-Decoder Example



RNN Recurrent Neural Network
CNN Convolutional Neural Network

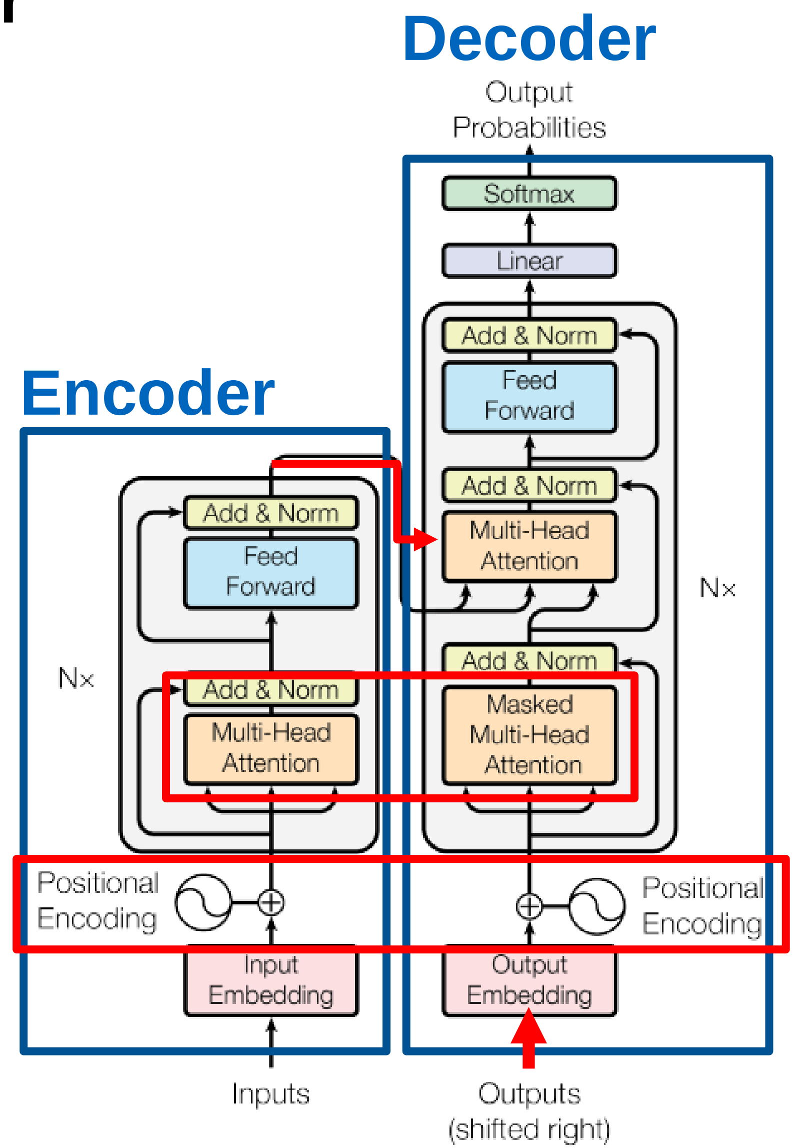
b. Deep Learning – Transformer

Overview

- Encoder-Decoder based
- Attention mechanism to encode important parts of a sequence and their relation to the other entries in it
- Decoder uses embeddings as well as previously generated sequence as input
- No RNN in original design

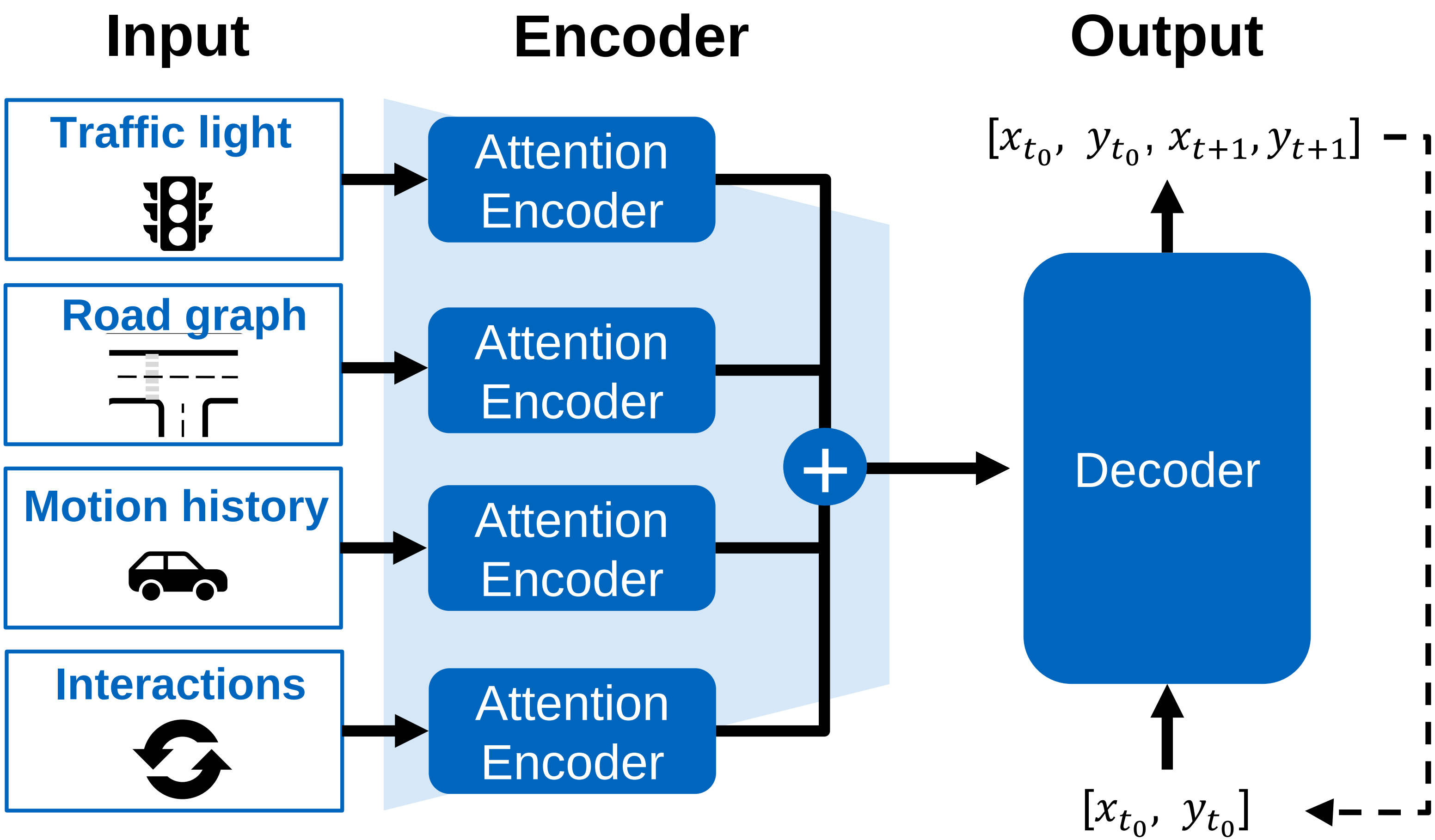
Applications

- Natural language processing
- Sequence analysis, e.g. amino acids
- Image generation (diffusion models)



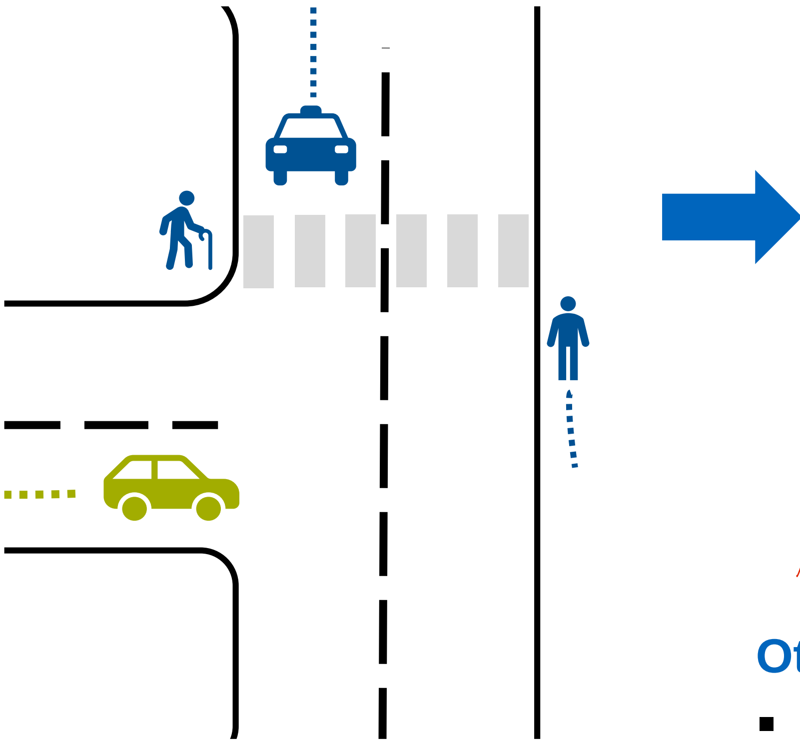
From “Attention is all you need” [72]

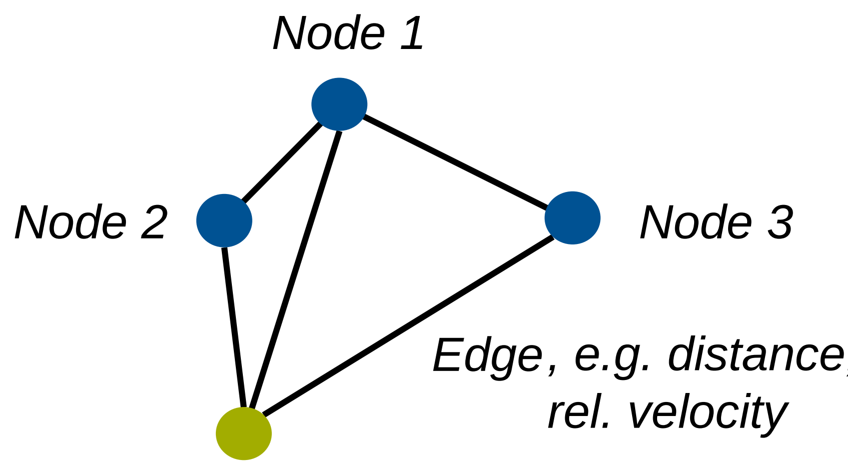
b. Deep Learning – Transformer Example



From “Wayformer” [73]

b. Deep Learning – Graph Neural Networks





Node 4 $= [\mathrm{c},\ x_{t_0},\ y_{t_0},\ x_{t_{-1}}, y_{t_{-1}},\ \ldots]$

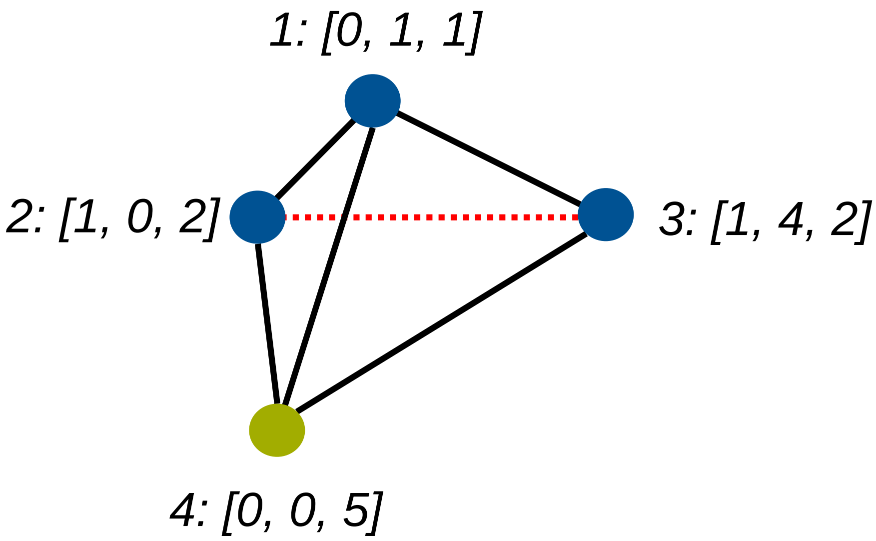
Node features

Other applications

- Graph or node classification
- Link prediction

c: class, e.g. pedestrian, car
x_{t_0}: x-position for t=0
y_{t_0}: y-position for t=0

b. Deep Learning – Graph Neural Networks Example



Adjacency matrix

N	1	2	3	4
1	1	1	1	1
2	1	1	0	1
3	1	0	1	1
4	1	1	1	1

[69], [70], [71]

b. Deep Learning – Graph Neural Networks Example

Example 1

1*0 + 1*1 + 1*1 + 1*0 = 2

Example 2

1*1 + 1*2 + 0*2 + 1*5 = 8

Node feature matrix

0	1	1
1	0	2
1	4	2
0	0	5

Adjacency matrix

1	1	1	1
1	1	0	1
1	0	1	1
1	1	1	1

Node embedding matrix

= Input feature matrix

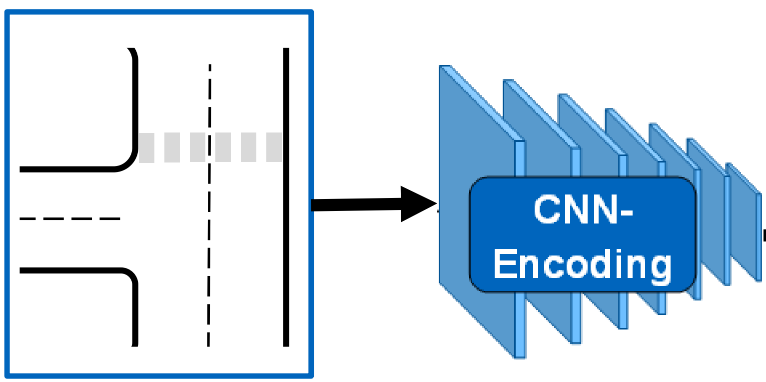
	5	10
1	1	
1	5	8
2	5	10

[69], [70], [71]

b. Deep Learning – Graph Neural Networks

Input feature matrix

2	5	10
1	1	8
1	5	8
2	5	10



RNN or CNN

Output,
e.g. relative position

x_{t+1}	y_{t+1}
1	2
3	2
4	1
2	2

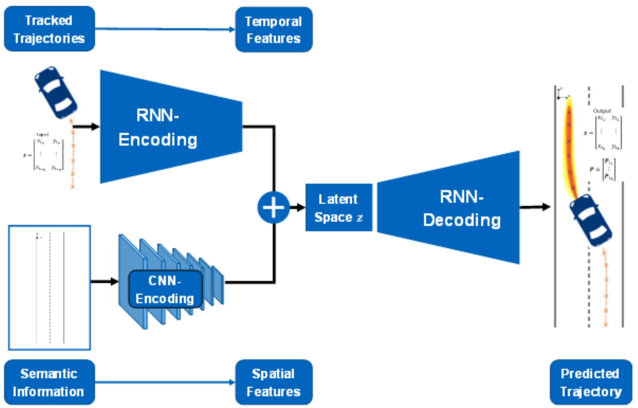
[69]

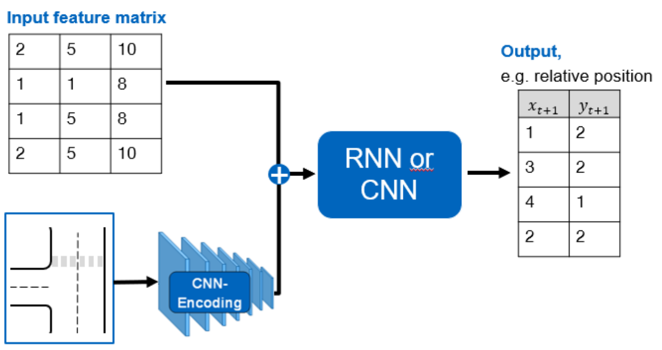
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network

b. Deep Learning - Summary

+ Modular, individual network design
+ Comprehensive prediction models: consideration of spatial information and interaction possible
+ Trajectory prediction and uncertainty quantification

- No explainability for safety verification
- Robustness of extrapolation to unknown data not defined





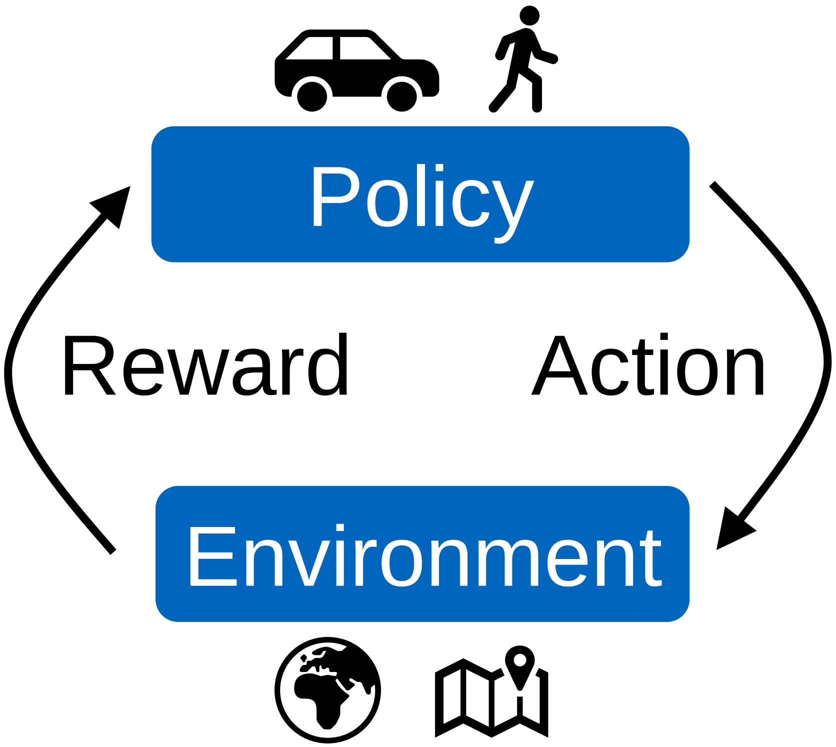
c. Inverse Reinforcement Learning - Motivation

Idea

- Better predictions by learning from observed, human behavior
- Instead of defining a reward function, approximate it from data
- Use reward function afterwards to determine optimal policy

Goal

- Approximate the reward function $R(s, a, s')$ to receive the maximal reward for every observed policy



c. Inverse Reinforcement Learning - Background

Input data

Set of trajectories $\{\xi\}$, which are assumed to be part of the optimal policy π^*

Given:

$$\{\xi\}_{i=1}^{N}, \xi_i = \{(s_t, a_t)\}_{t=1}^{T}; a_t \sim \pi^*(s_t)$$

Reward function

- Reward function R is linear combination of L features ϕ_i
- Initialize the feature weights w_i randomly

Initialization:

$$R(s) = w_0\phi_0(s) + \cdots + w_L\phi_L(s)$$

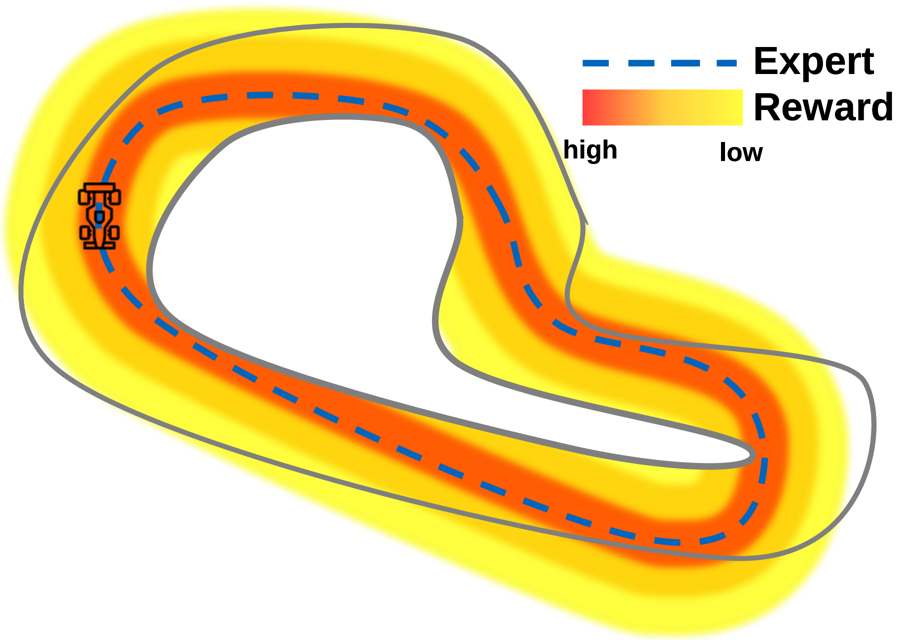
Training process

- Through Maximum Entropy IRL

c. Inverse Reinforcement Learning - Summary

- \+ Robust, general and transferable
- \+ Learning directly from observed data

- \- No direct policy output
- \- Hard to train in environments with sparse rewards / no direct reward function at all
- \- Features need to be defined



Prediction
Prof. Dr. Markus Lienkamp

Dipl.-Ing. Nico Uhlemann

Agenda

1. Foundations
2. Knowledge-Based Prediction
    a. State Estimation
    b. Reachable Sets
3. Learning-Based Prediction
    a. Clustering and Classification
    b. Deep Learning
    c. Inverse Reinforcement Learning
4. **Summary and Outlook**

Knowledge-based Prediction

State Estimation

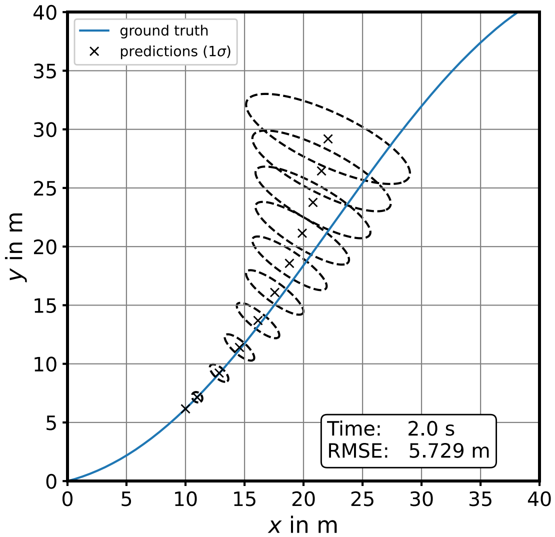
- Kinematic Models
- Bayesian Filter for Probabilistic Trajectory Prediction

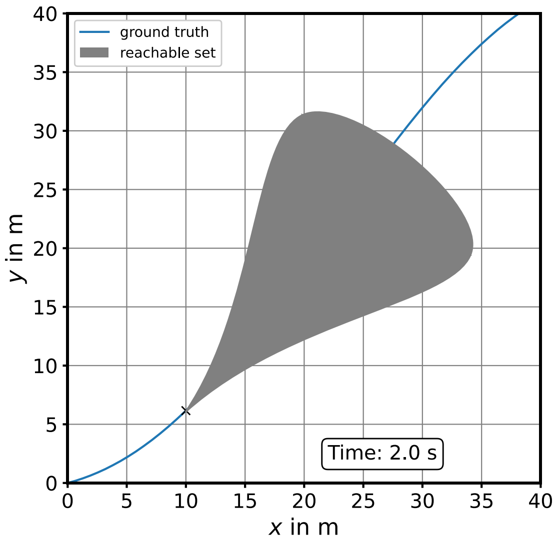
Reachability Analysis

- Coverage of all possible states within the dynamic limits of the object

Robust algorithms

- High accuracy in short-term prediction
- No comprehensive trajectory prediction





Learning-based Prediction

Clustering and Classification

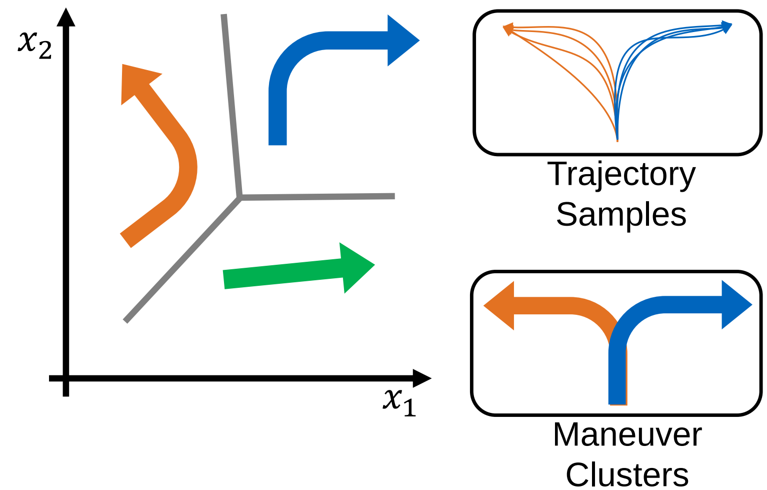
- Maneuver Prediction
- Structured Environments

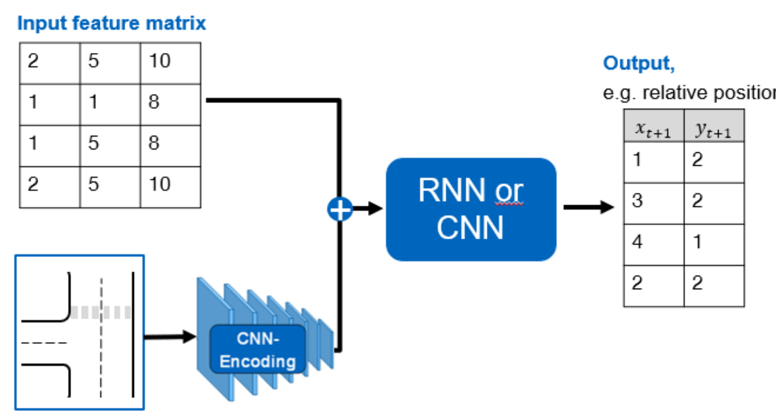
Deep Learning Algorithms

- Encoder-Decoder
- Transformer
- Graph Neural Networks

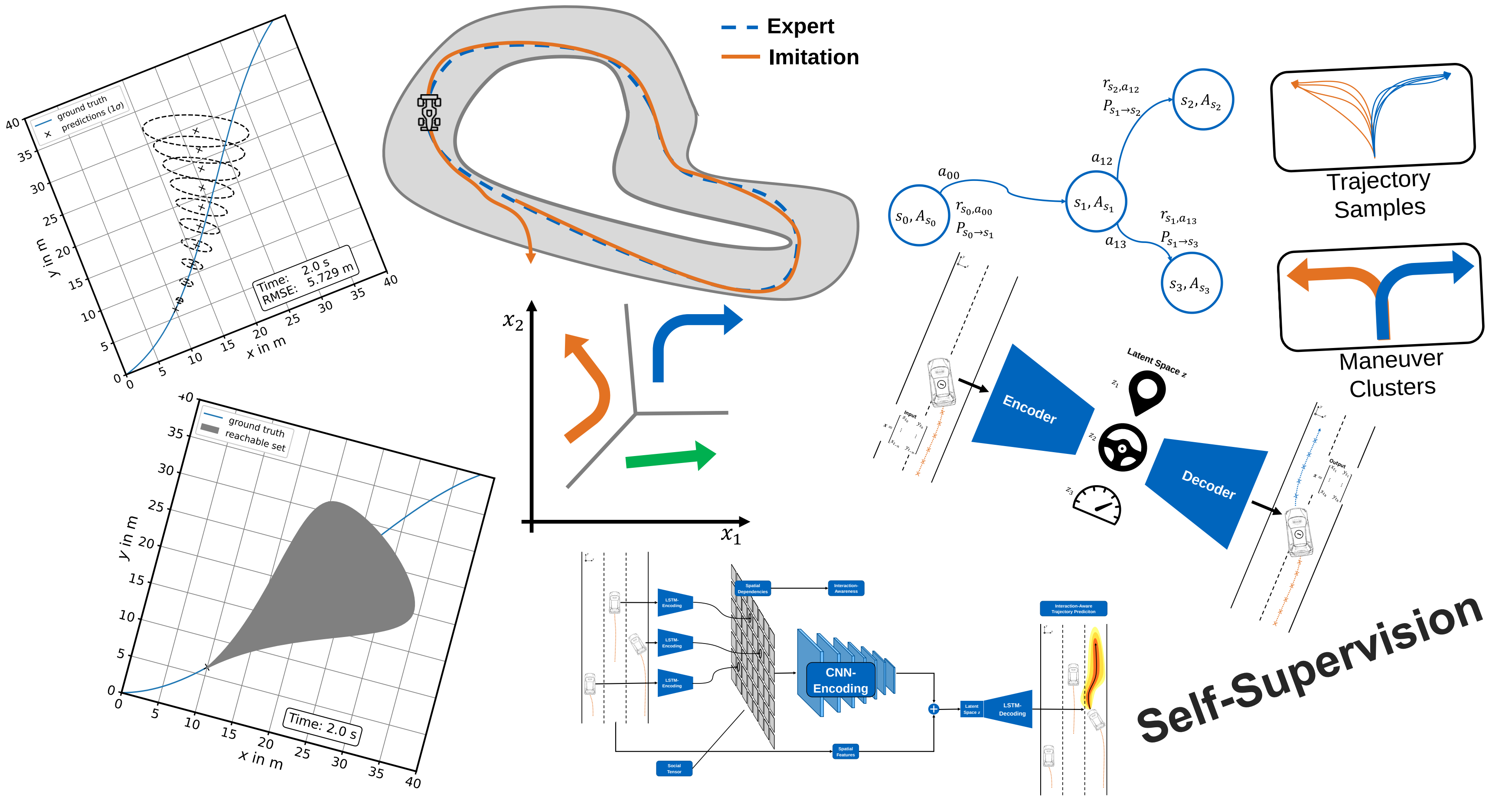
Reinforcement Learning

- Reward function approximation





Outlook – Motion Prediction is an open research topic



Outlook

Interactive, dynamic Planning

Prediction is essential for dynamic planning in complex environment and Input to interactive planning concepts (Game Theory etc.).

Scenario Understanding

Comprehensive prediction aims to encode the human driving behavior

Learning from Demonstration

Prediction could be the enabler for new planning algorithms

→ **Chapter 11: End-to-End**