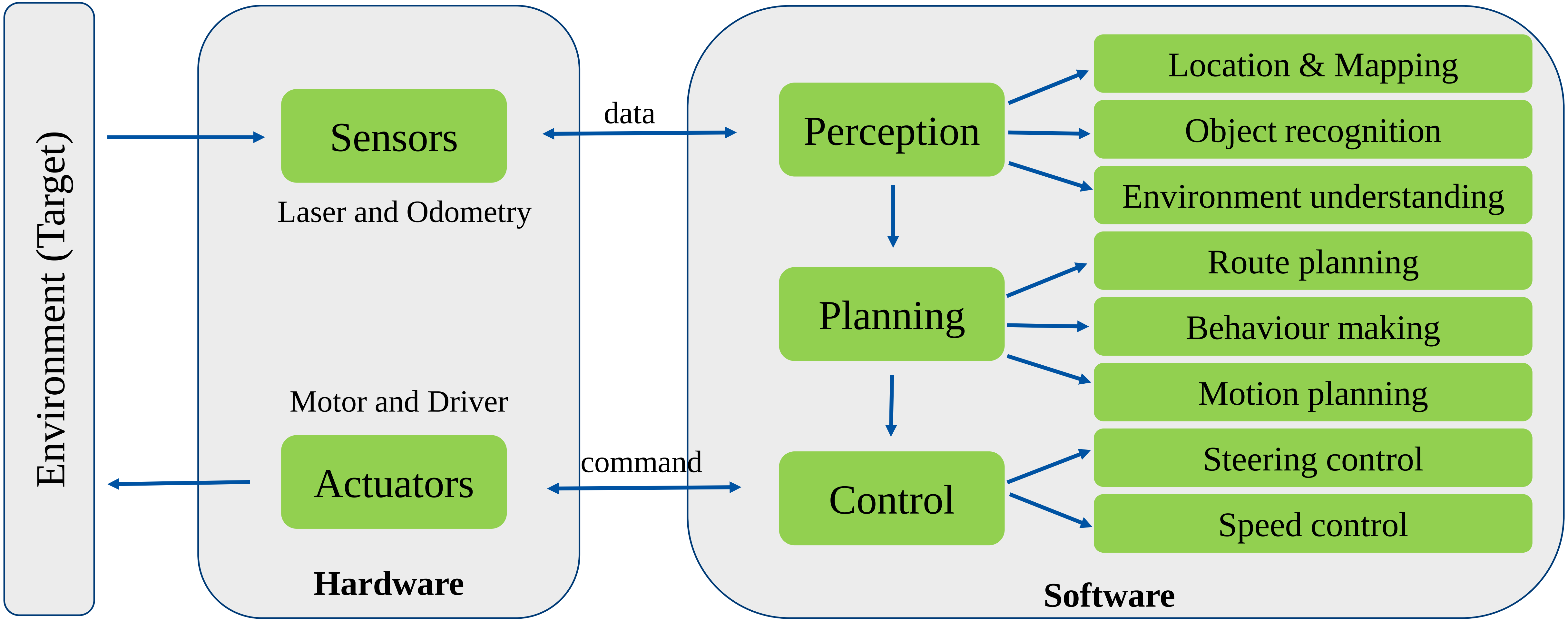
Autonomous Driving Framework Using Traditional Method

Benchun Zhou

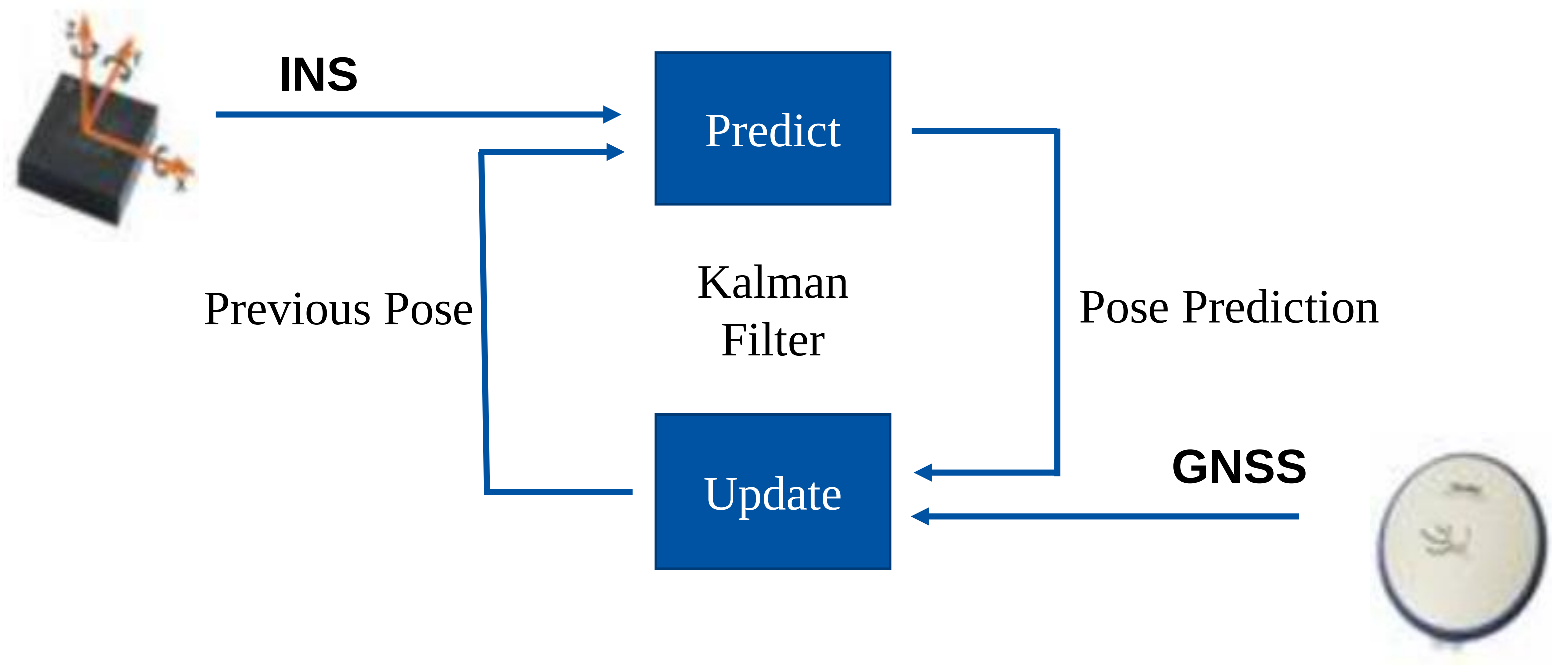
School of Automation Science and Electronic Engineering
Beihang University

- **Framework**

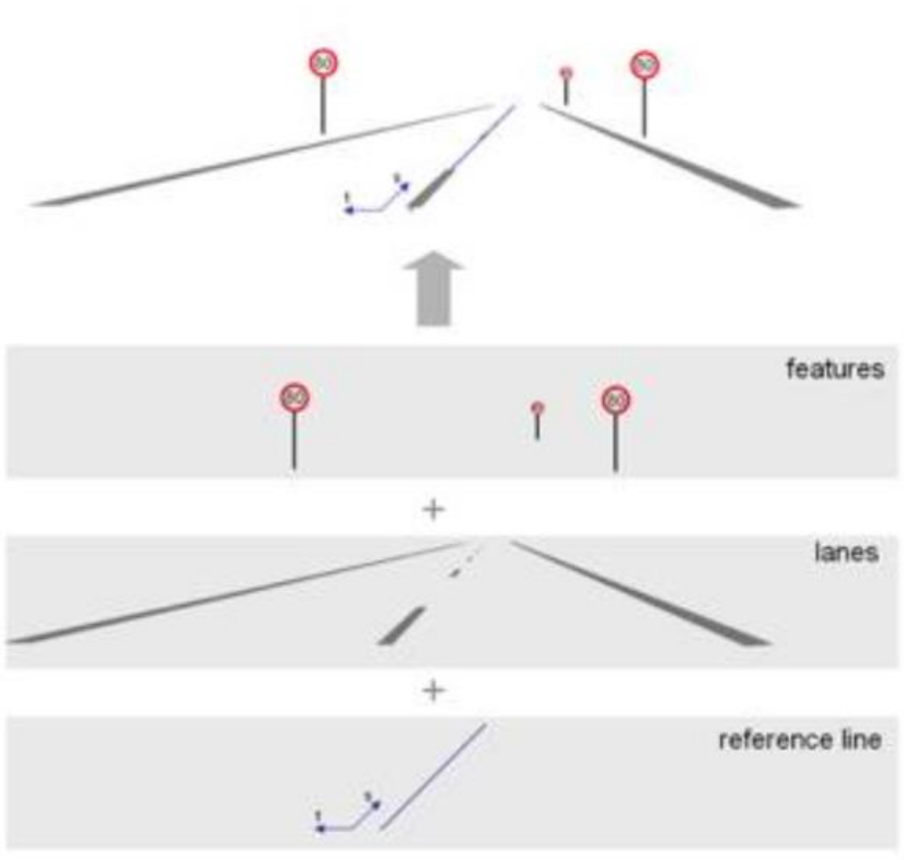

Environment (Target)
Sensors
Laser and Odometry
data
Motor and Driver
Actuators
command
Hardware
Perception
Planning
Control
Location & Mapping
Object recognition
Environment understanding
Route planning
Behaviour making
Motion planning
Steering control
Speed control
Software

- **Localization & Mapping: GNSS/INS**

- **GNSS:** accurate but low frequency
- **INS:** high frequency but inaccurate
- **Kalman Filter:** get the best from both



- **Localization & Mapping: LiDAR & HD Map**



- **Layer Map:**
- Foundation layer is a 2D grid map with 5cm by 5cm resolution
- Road reference line is then added
- Next layer is the lane information
- Other semantic feature are then added

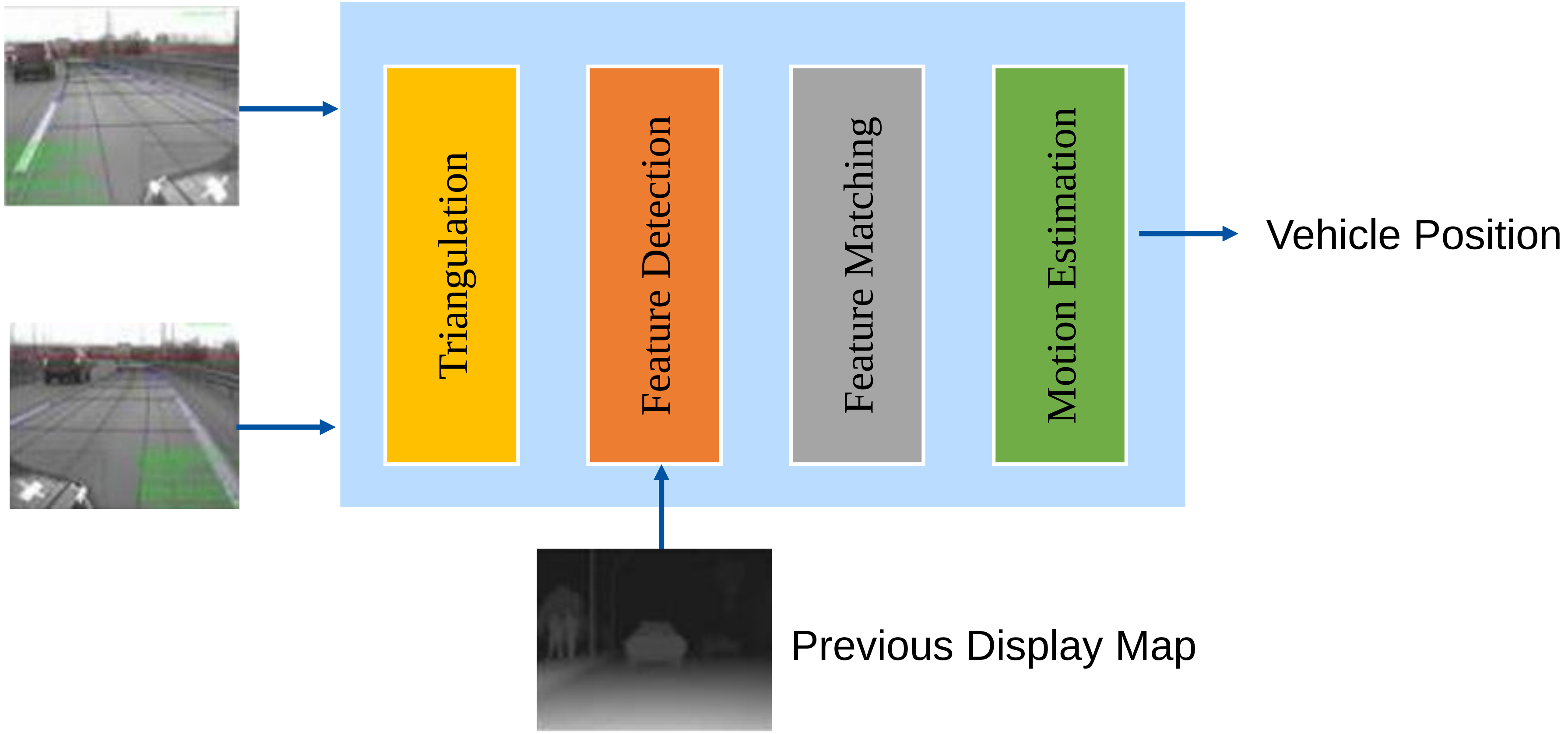
- **Requirements**
- Fresh
- Precise
- Integration with the client systems

• Localization & Mapping: Visual Odometry

- Stereo visual odometry
- Mono visual odometry
- Visual Inertial odometry

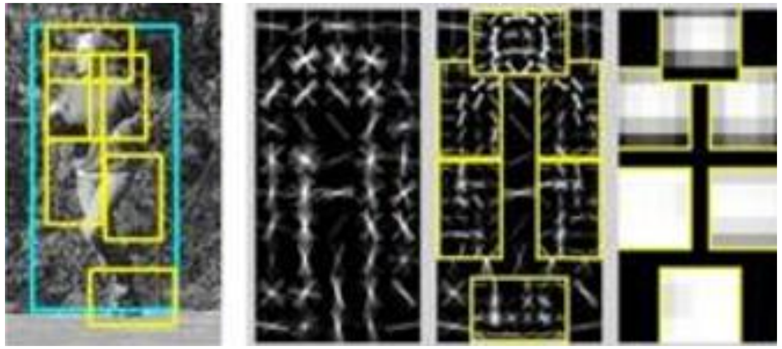
Visual Odometry Pipeline

Stereo Image

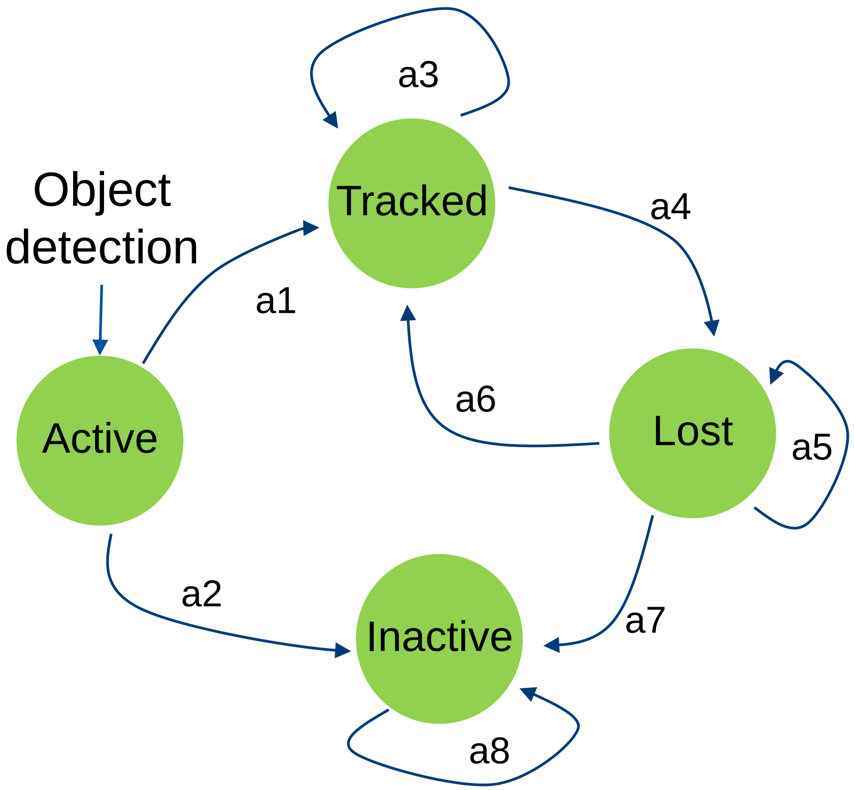


- **Object Recognition: Road Detection**

Image Input → Normalize gamma & color → Compute gradients → Weight vote into spatial & orientation cell → Contract normalize over overlapping spatial bloks → Collect HOG's over detection window → Linear SVM → Person/ classification



- **Object Recognition: Moving Object Recognition**



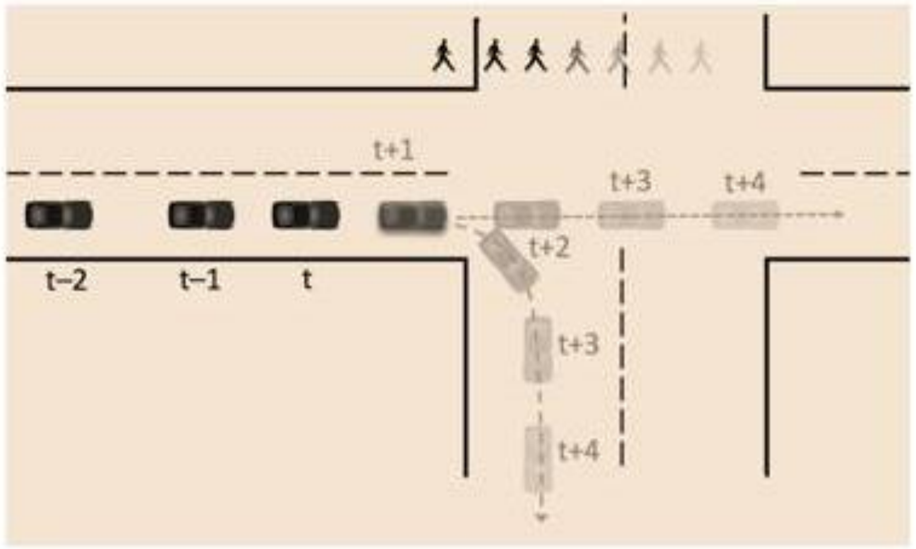
1. Perception

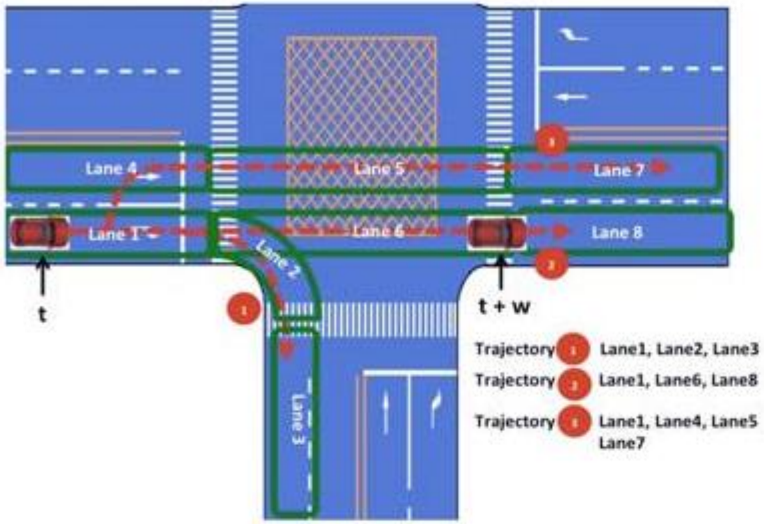
- **Environment Understanding**

- **Metric dimension:** an environment's geometric layout
- **Semantic Mapping:** the semantic meanings of different places
- **Activity Learning:** the activity patterns of human agents living in it

- **Traffic Prediction**

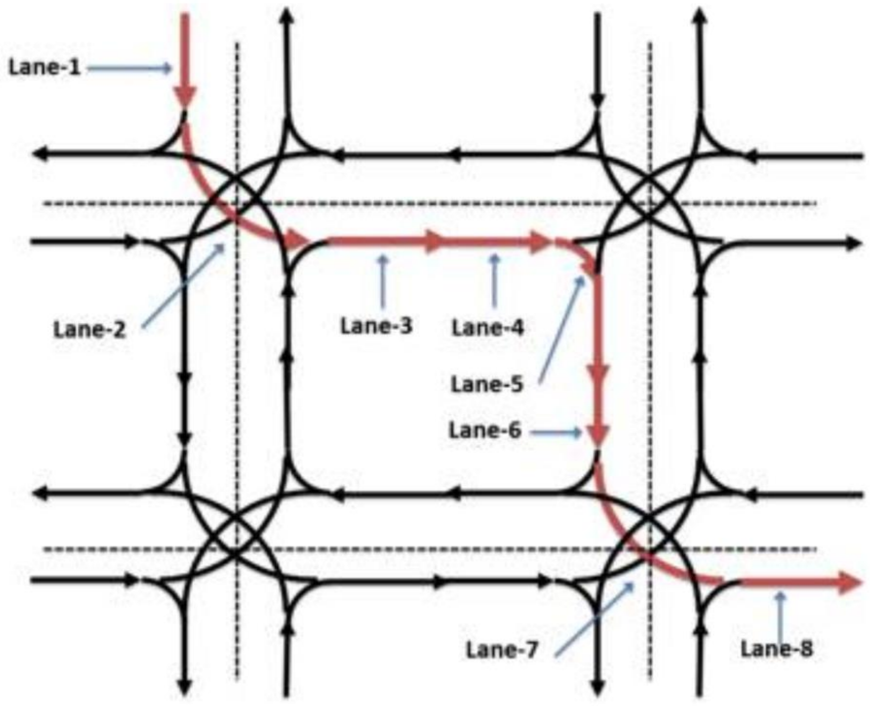
- Classification problem for categorical road object behaviors
- Regression problem for generating the predicted path with speed and time info

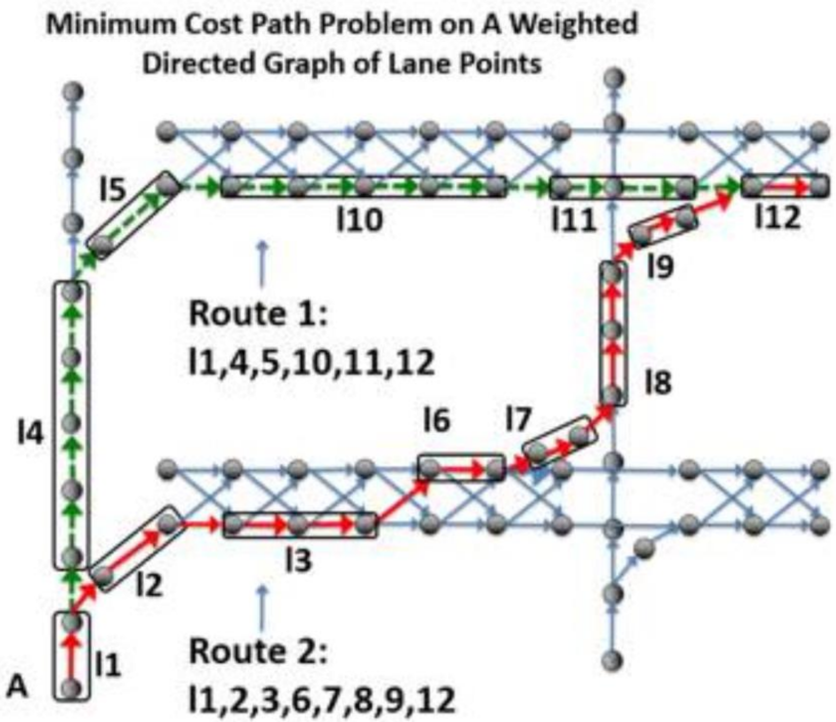




- **Route Planning**

- Modeled as weight directed graphs
- Short path problem: Dijkstra and A star

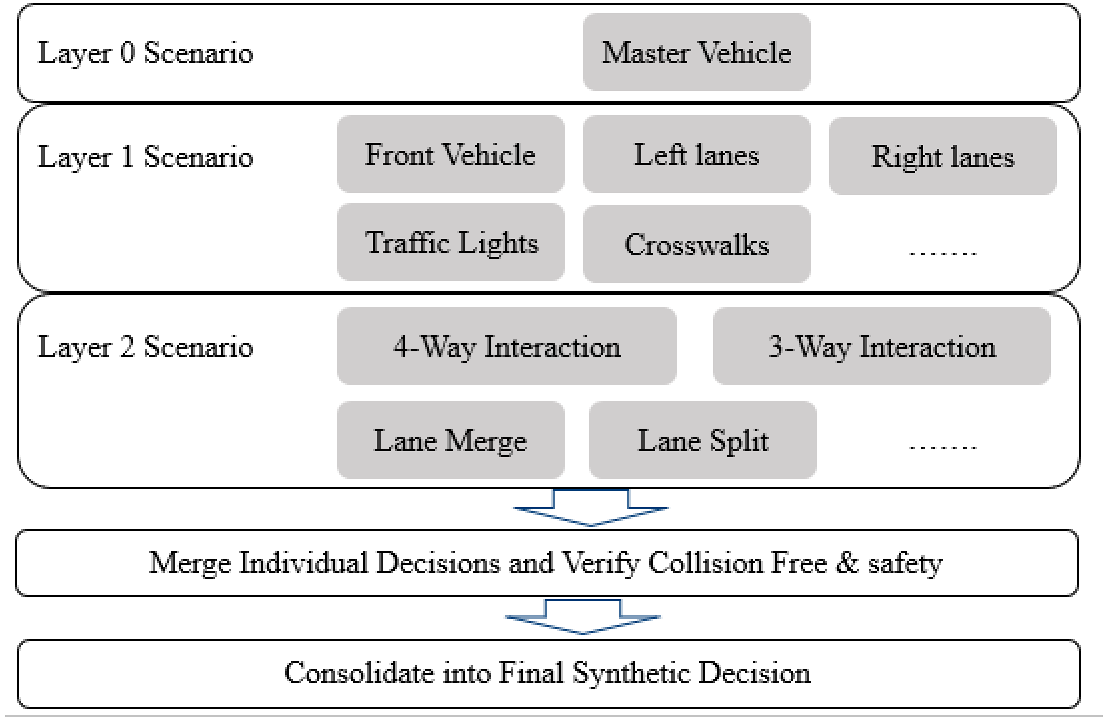


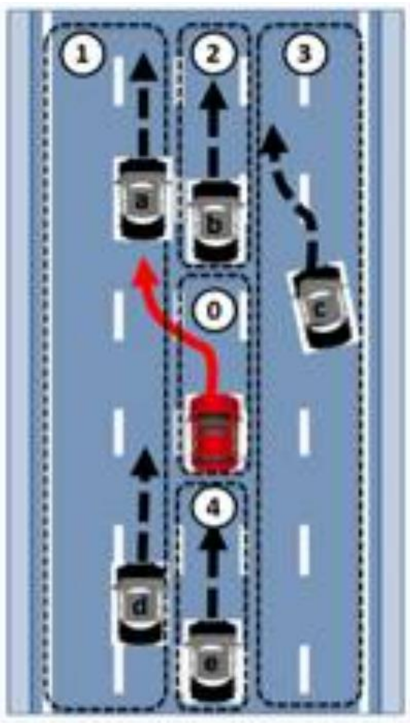


2. Planning

- **Behavior planning**

- Ruled-based “drive-and-conquer” approach: layer
- Markov Decision Process
- Synthetic decision and individual decision

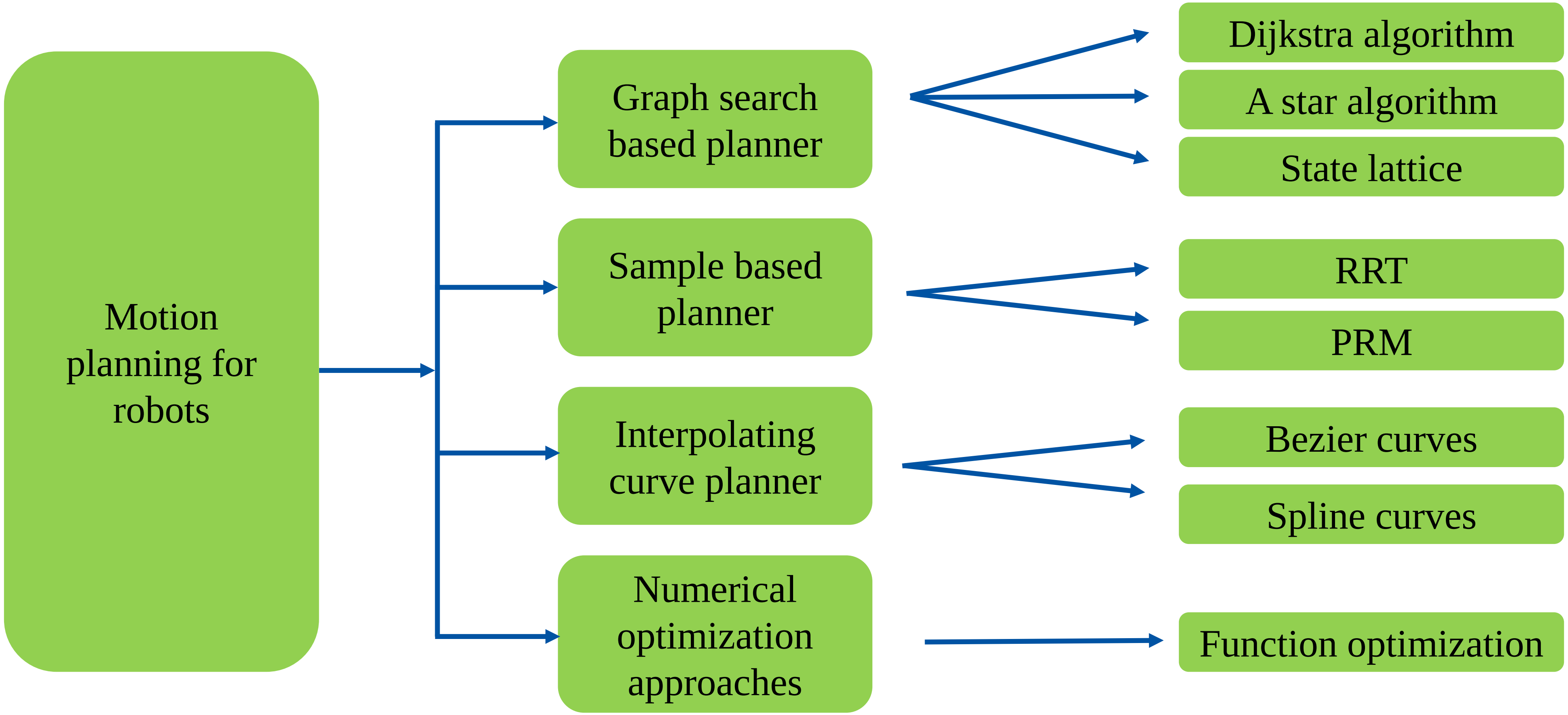




- **Synthetic Decision**
- Switch lane from current lane to the left lane: yield vehicle a, overtake vehicle a, and attention to vehicle b at current lane
- **Scenarios and Individual Decisions**
- 0. Master Vehicle
- 1. Left Line(s)
- 2. Front Vehicle(s)
- 3. Right Vehicle(s)
- 4. Rear vehicle(s)

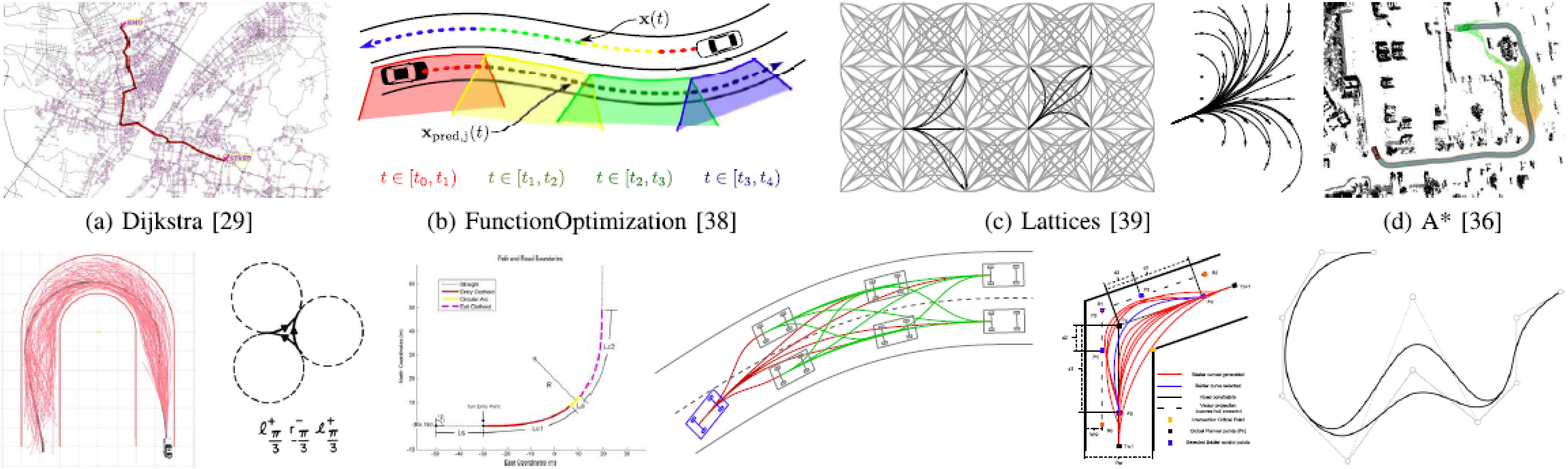
- **Motion Planning**

- Motion planning breaks down a desired movement task into discrete motions that satisfy movement constraints and possibly optimizes some aspect of the movement.



• Motion Planning

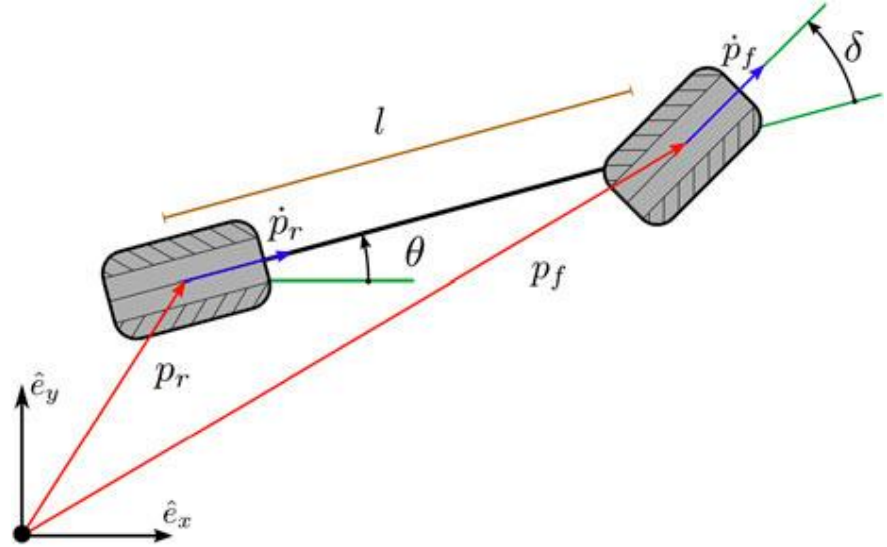
• Motion planning breaks down a desired movement task into discrete motions that satisfy movement constraints and possibly optimizes some aspect of the movement.



(a) Dijkstra [29] (b) FunctionOptimization [38] (c) Lattices [39] (d) A* [36]

(e) RRT [40] (f) Line&Circle [41] (g) Clothoid [42] (h) Polynomial [43] (i) Bezier [44] (j) Spline [45]

- **Feedback Control:** Kinematic Model



- Velocity of rear wheel (X_r, Y_r)

$$v_r = \dot{X}_r \cos\varphi + \dot{Y}_r \sin\varphi$$

- Constraints of front and rear wheel

$$\begin{cases} \dot{X}_f \sin(\varphi+\delta_f) - \dot{Y}_f \cos(\varphi+\delta_f) = 0 \\ \dot{X}_r \sin\varphi - \dot{Y}_r \cos\varphi = 0 \end{cases} \Rightarrow \begin{cases} \dot{X}_r = v_r \cos\varphi \\ \dot{Y}_r = v_r \sin\varphi \end{cases}$$

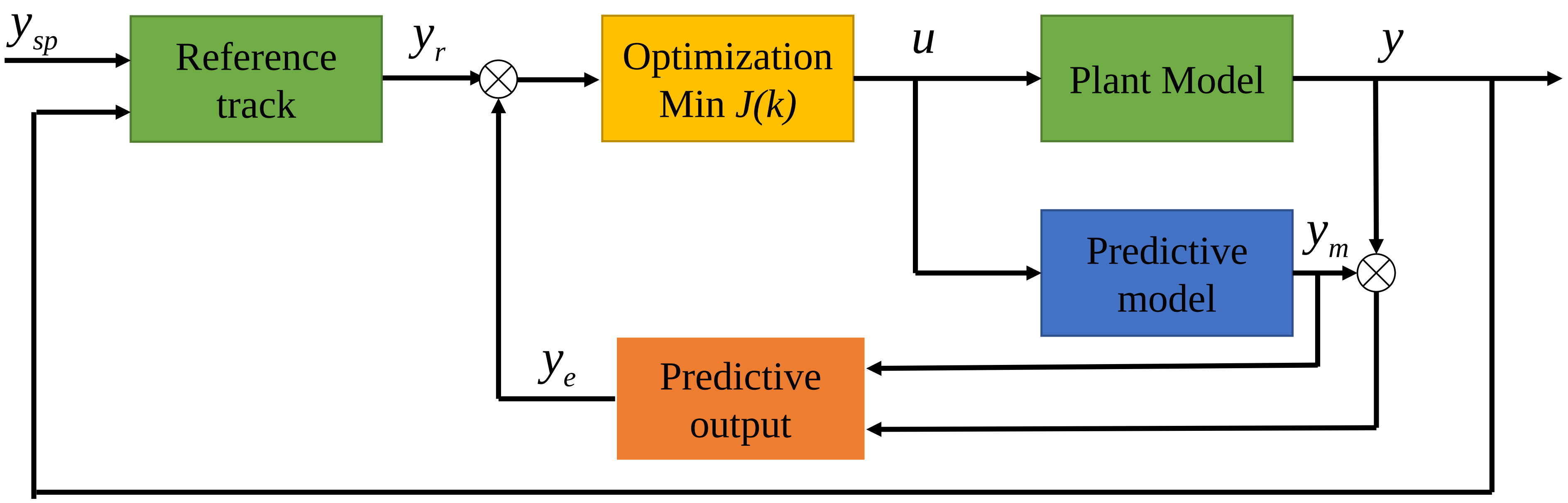
- Geometry of the front and rear wheel

$$\begin{cases} X_f = X_r + l\cos\varphi \\ Y_f = Y_r + l\sin\varphi \end{cases} \Rightarrow \omega = \frac{v_r}{l}\tan\delta \qquad \begin{cases} R = v_r/\omega \\ \delta_f = \arctan(l/R) \end{cases}$$

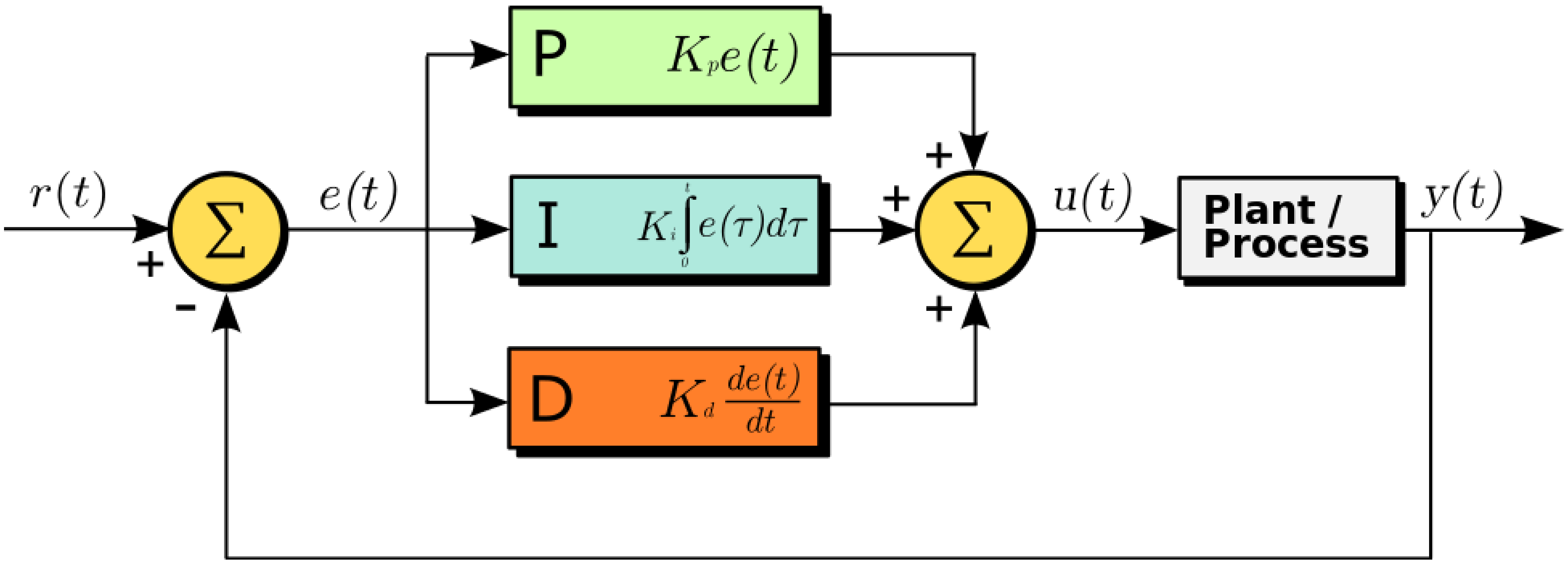
- Kinematic model

$$\begin{bmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} \cos\varphi \\ \sin\varphi \\ \tan\delta/l \end{bmatrix} v_r$$

- **Feedback Control:** Model Predictive Control



- **Feedback Control:** PID controller



THANKS YOU!