

Autonomous Driving: Planning, Control & Other Topics

Jan 8th, 2018

Sahil Narang

University of North Carolina, Chapel Hill



Autonomous Driving: Main Components

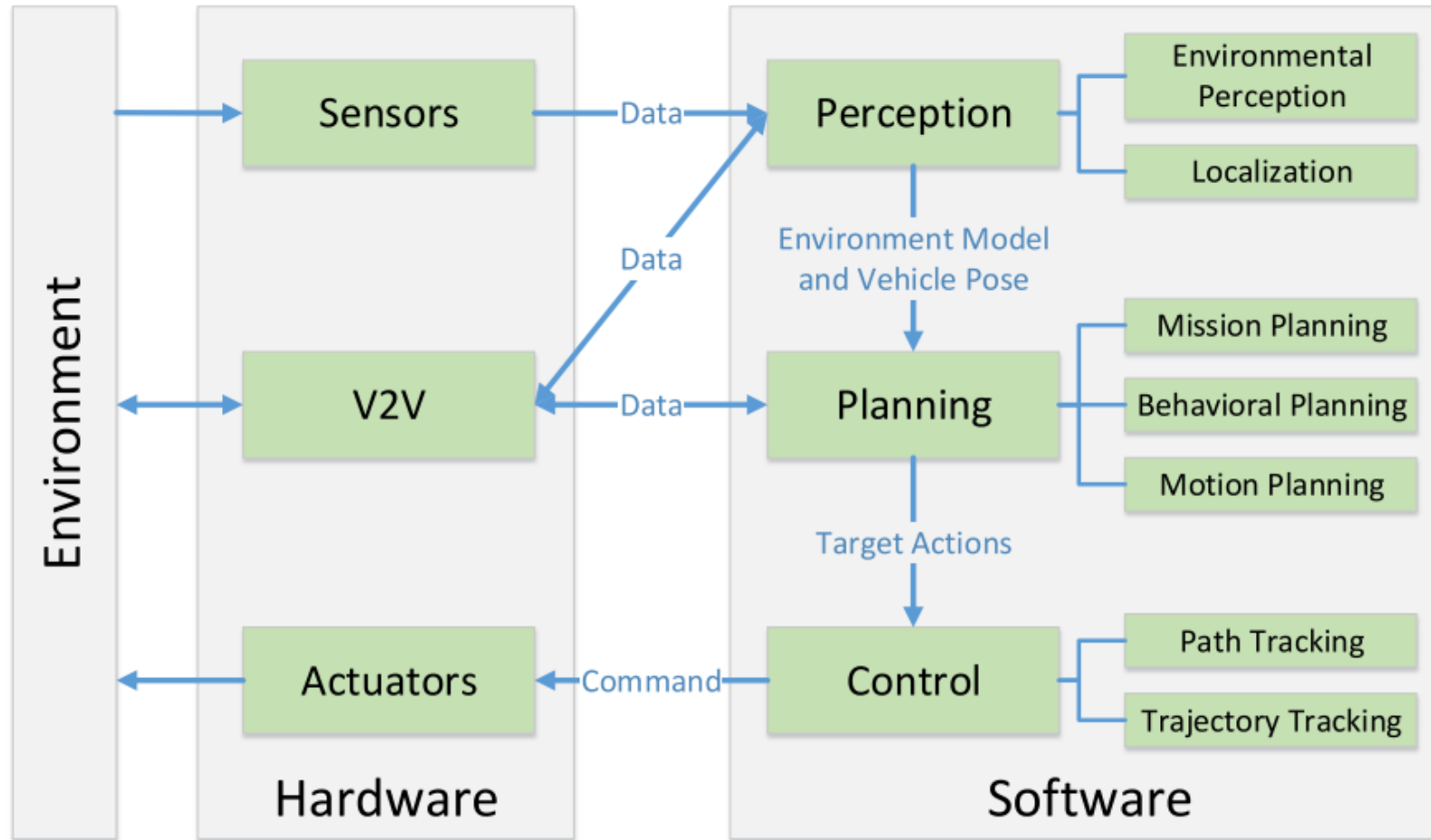


Figure 2. A typical autonomous vehicle system overview, highlighting core competencies.



LIDAR Interference

- ★ Probability of LIDAR interference is very low
 - ⑩ LIDAR uses **very focused** light pulses
 - ⑩ Broad range of “Pulse Repetition Frequency”
 - ⑩ Most approaches fuse data from from various sensors
- ★ Kim, G., Eom, J., & Park, Y. (2015, June). Investigation on the occurrence of mutual interference between pulsed terrestrial LIDAR scanners. In *Intelligent Vehicles Symposium (IV), 2015 IEEE* (pp. 437-442). IEEE.



Autonomous Driving: Main Components

★ Perception

- ⑩ collect information and extract relevant knowledge from the environment.

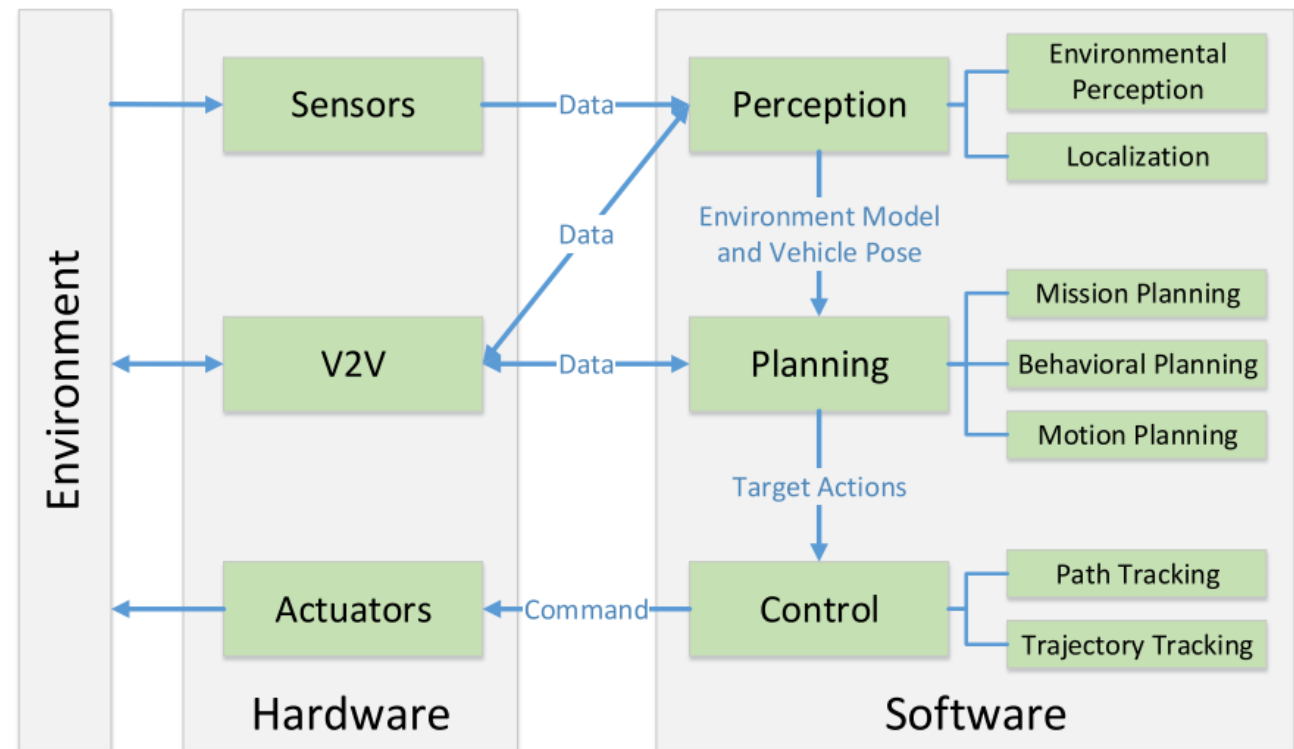


Figure 2. A typical autonomous vehicle system overview, highlighting core competencies.



Autonomous Driving: Main Components

★ Planning

- ⑩ Making purposeful decisions in order to achieve the robot's higher order goals

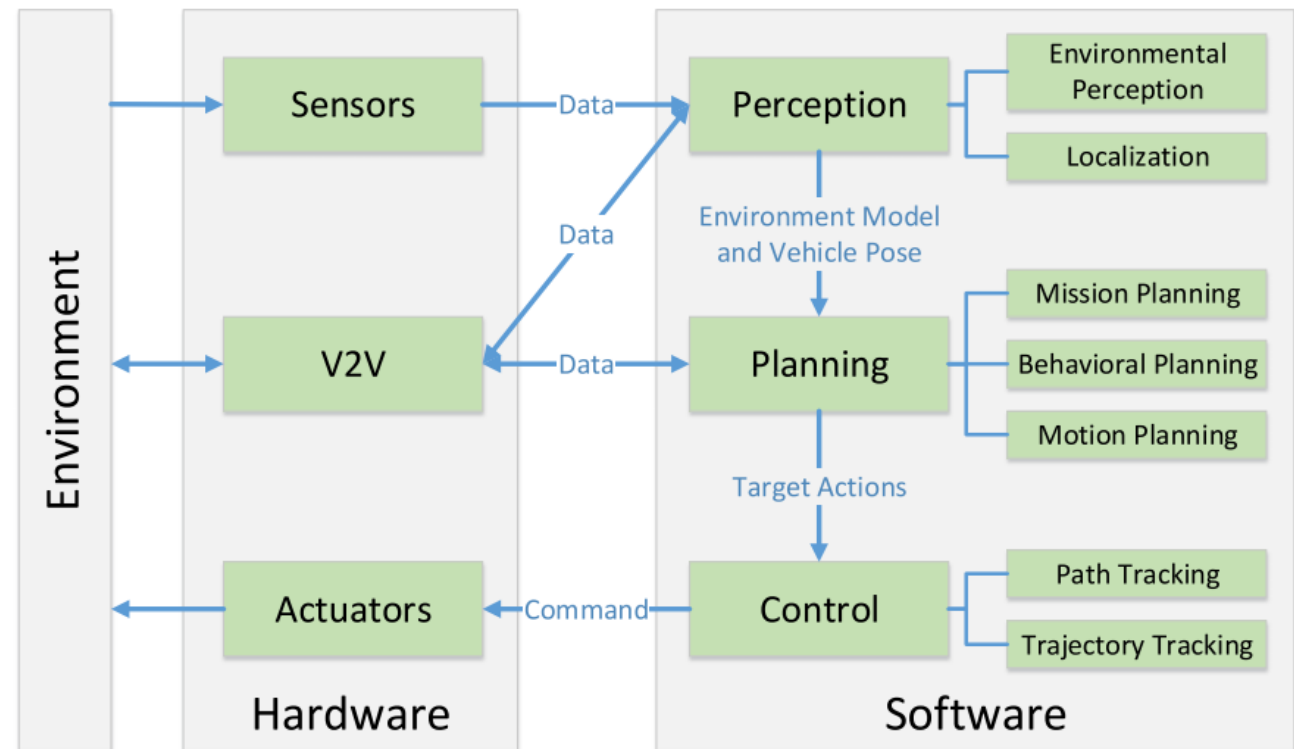


Figure 2. A typical autonomous vehicle system overview, highlighting core competencies.



Autonomous Driving: Main Components

★ Control

⑩ Executing planned actions

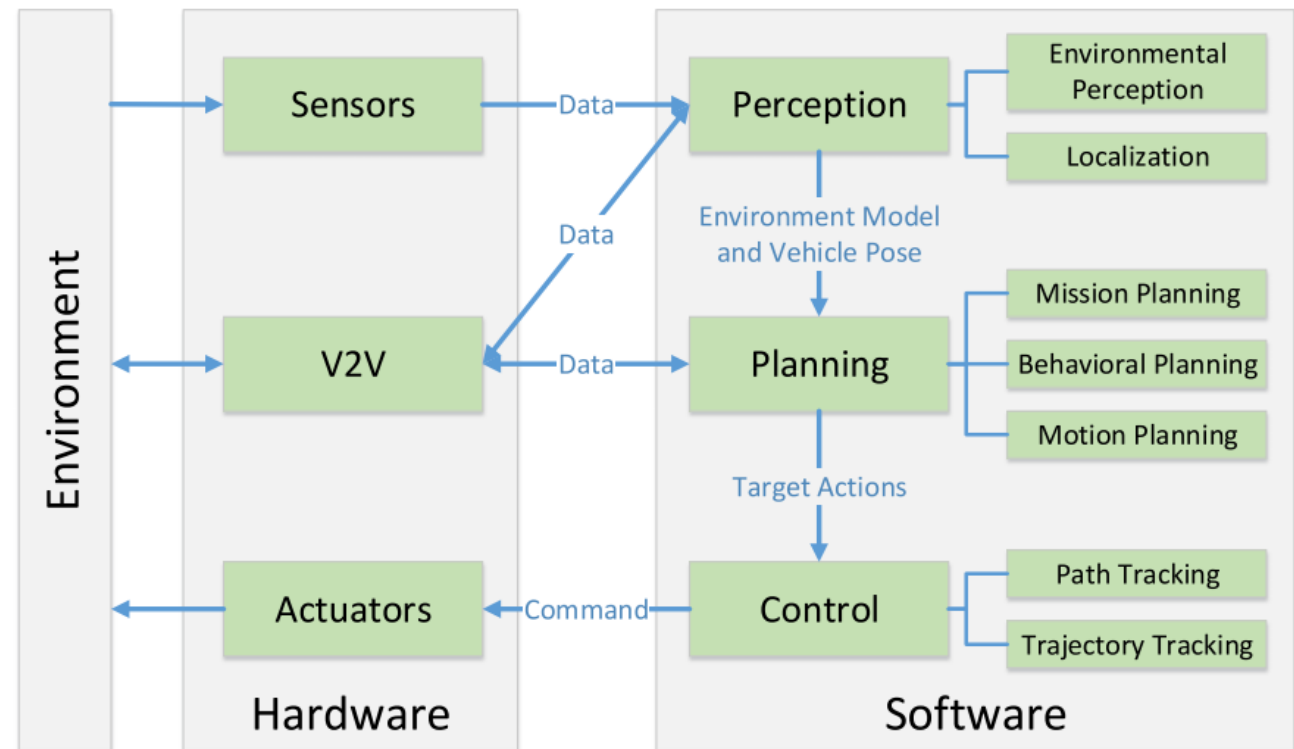


Figure 2. A typical autonomous vehicle system overview, highlighting core competencies.



Structure

- ★ Modeling a car
 - ⑩ State Space
 - ⑩ Kinematic constraints
 - ⑩ Dynamic constraints
- ★ Motion Planning
- ★ Control
- ★ Interaction with other Drivers
- ★ Case Studies



Autonomous Driving: State Space

- ★ “The set of attribute values describing the condition of an autonomous vehicle at an instance in time and at a particular place during its motion is termed the ‘state’ of the vehicle at that moment”
- ★ Typically a vector with position, orientation, linear velocity, angular velocity
- ★ **State Space**: set of all states the vehicle could occupy

Autonomous Driving: State Space

★ Examples:

⑩ 3D space with velocity

★ $(p_x, p_y, p_z, \theta_x, \theta_y, \theta_z, v_x, v_y, v_z, \omega_x, \omega_y, \omega_z)$

★ $(\vec{p}, \vec{\theta}, \vec{v}, \vec{\omega})$

⑩ 2D space with acceleration

★ $(p_x, p_y, \theta, v_x, v_y, \omega, a_x, a_y, \alpha)$

★ $(\vec{p}, \theta, \vec{v}, \omega, \vec{a}, \alpha)$



Autonomous Driving: State Space

★ Examples:

⑩ 2D space with blinker booleans

★ $(\vec{p}, \theta, \vec{v}, \omega, bl_l, bl_r)$

⑩ State contains everything we need to describe the robot's current configuration!

⑩ Neglect some state variables when planning



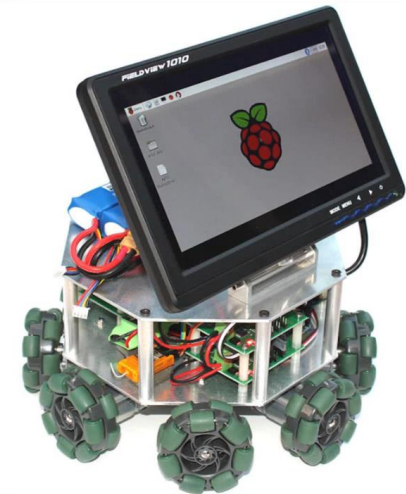
Autonomous Driving: Holonomicity

★ “Holonomic” robots

- ⑩ Holonomic system are systems for which all constraints are integrable into positional constraints.
- ⑩ Holonomic system where a robot can move in any direction in the configuration space.
- ⑩ Controllable DOF = total DOF

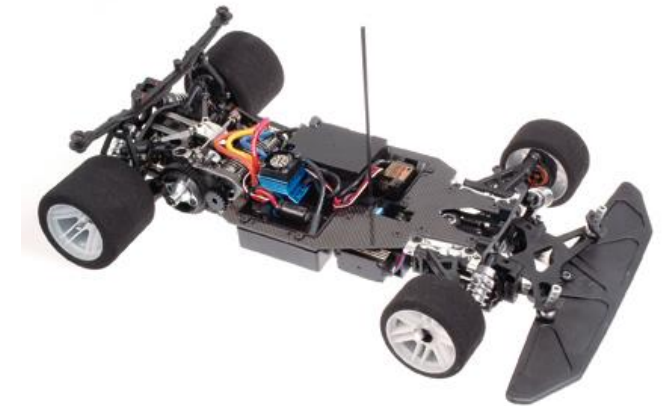
★ Examples:

- ⑩ Omni-drive base
- ⑩ <https://youtu.be/9ZCUxXajzXs>



Autonomous Driving: Holonomicity

- ★ Cars are “non-holonomic” robots
 - ⑩ Typically 3 values describing C-Space
 - ★ x, y, θ
 - ⑩ 2 “kinematic” constraints
 - ★ Can only move forward or backward, tangent to body direction
 - ★ Can only steer in bounded radius



Kinematic Constraints

★ Kinematics of Motion

- ⑩ “the branch of mechanics that deals with pure motion, without reference to the masses or forces involved in it”
- ⑩ Equations describing conversion between control and motion
- ⑩ Control: inputs to the system
 - ★ In vehicle: steering and throttle
 - ★ Also referred to as “Action” in literature



Autonomous Driving: Holonomicity

- ★ kinematic and dynamic constraints can be considered “rules” governing the state evolution function
- ★ For state $s_t \in S$, control input $u_t \in U$, time $t \in T$:
 - ⑩ $F(s_t, u_t, \Delta t) \rightarrow s_{t+1}$
- ★ Ex:
 - ⑩ A car cannot turn in place. No amount of steering will accomplish this
 - ⑩ A Roomba can turn in place



Kinematic Constraints

★ Kinematic models of a car

⑩ Single-track Bicycle (or simple car model)

★ 3-DOF configuration: (x, y, θ)

★ 2-DOF control: steering (u_ϕ) , speed (u_s)

★ Full state: $(x, y, \theta, u_s, u_\phi, L)$

⑩ Equations of motion:

$$\dot{x} = u_s * \cos(\theta) \quad \dot{y} = u_s * \sin(\theta)$$

$$\dot{\theta} = \frac{u_s}{L} * \tan(u_\phi)$$

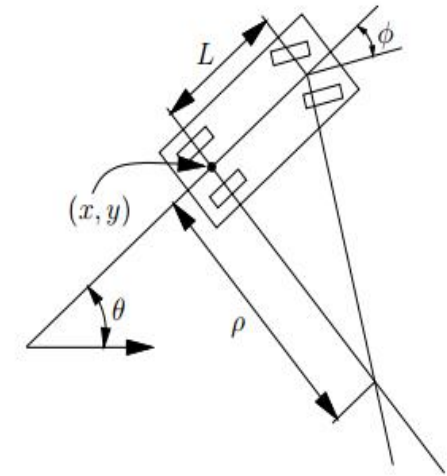


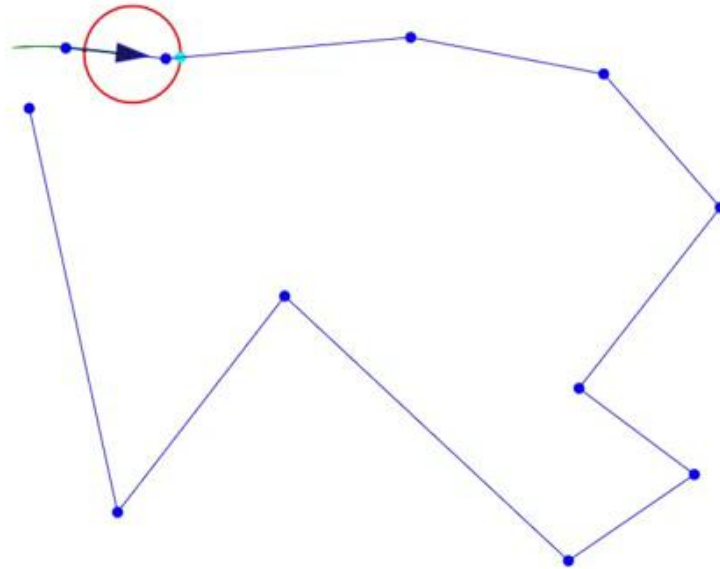
Figure 13.1: The simple car has three degrees of freedom, but the velocity space at any configuration is only two-dimensional.

Kinematic Constraints

★ Kinematic models of a car

⑩ Single-track Bicycle example

⑩ <https://www.youtube.com/watch?v=TyW1BPpHy18>



Kinematic Constraints

- ★ Kinematic models of a car
 - ⑩ Extended Car w. linear integrators
 - ⑩ Assume speed is fixed to 1
 - ⑩ 6-DOF configuration $(x, y, \theta, \phi, \omega)$
 - ★ 1-DOF Control:
 - Angular acceleration (u_a)
 - ★ Full state $(x, y, \theta, v, \phi, \omega, u_s, u_v, L)$



Kinematic Constraints

★ Extended Car w. linear integrators

⑩ Equations of motion

$$\begin{aligned} \star \dot{p}_x &= \cos(\theta) & \dot{p}_y &= \sin(\theta) \\ \dot{\theta} &= \frac{\tan(\phi)}{L} & \dot{\phi} &= \omega & \dot{\omega} &= \mu_\alpha \end{aligned}$$

⑩ Steering is continuous C^1

⑩ Control is more complex



Kinematic Constraints

★ Example: Stopping the car

⑩ Let u_s and u_v represent speed and velocity controls

⑩ Simple-car: $u_s = 0$

⑩ LI-car $u_v = -v$ iff $\max(u_v) \geq v$ *else* $\max(U_v)$

★ Car will not necessarily stop right away

★ Error increases as we increase the number of integrators



Dynamic Constraints

- ★ “the branch of mechanics concerned with the motion of bodies under the action of forces.”
- ★ Tires subject to lateral and longitudinal force during steering / accelerating
 - ⑩ If lateral force exceeds friction force
 - ★ Fishtailing
 - ⑩ If longitudinal force exceeds friction force
 - ★ Skidding



Dynamic Constraints

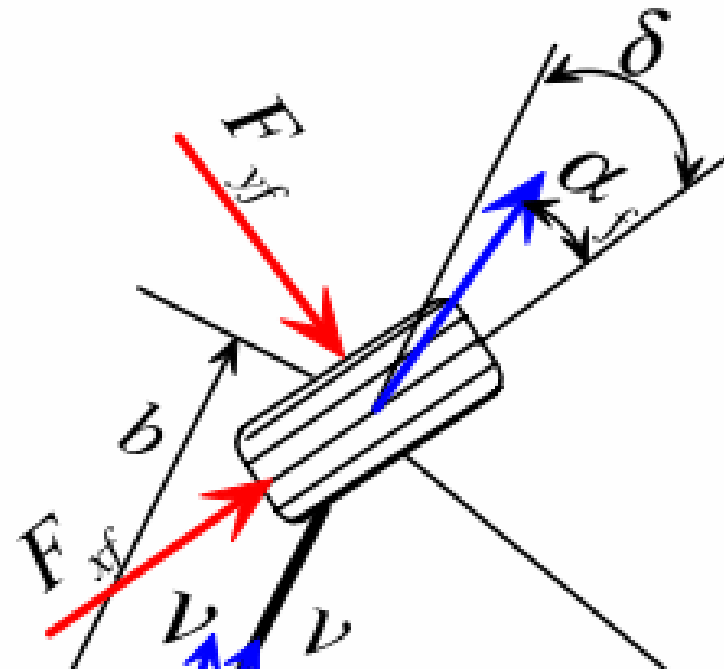
- ★ No longer directly control acceleration and steering
 - ⑩ Apply engine force
 - ⑩ Apply steering force
- ★ Diminishing returns on each force at limits of control



Dynamic Constraints

★ Dynamic Bicycle model with linear tires

- ⑩ F_y lateral force on tire
- ⑩ F_x longitudinal force on tire
- ⑩ α_f “slip angle” of tire
- ⑩ δ steering angle



Dynamic Constraints

★ Dynamic Bicycle model with linear tires

- ⑩ No load transfer between tires
- ⑩ Larger state space including tire stiffness

$$\begin{aligned}
 F_{xf} \cos \delta - F_{yf} \sin \delta + F_{xr} &= m(\dot{v}_x - v_y \dot{\psi}) \\
 F_{xf} \sin \delta + F_{yf} \cos \delta + F_{yr} &= m(\dot{v}_y + v_x \dot{\psi}) \\
 (F_{xf} \sin \delta + F_{yf} \cos \delta)b - F_{yr}c &= I_z \ddot{\psi} \\
 F_y &= C_\alpha \alpha
 \end{aligned}$$

- ★ F_x longitudinal force
- ★ F_y lateral force
- ★ m mass
- ★ I_z yaw moment of inertia

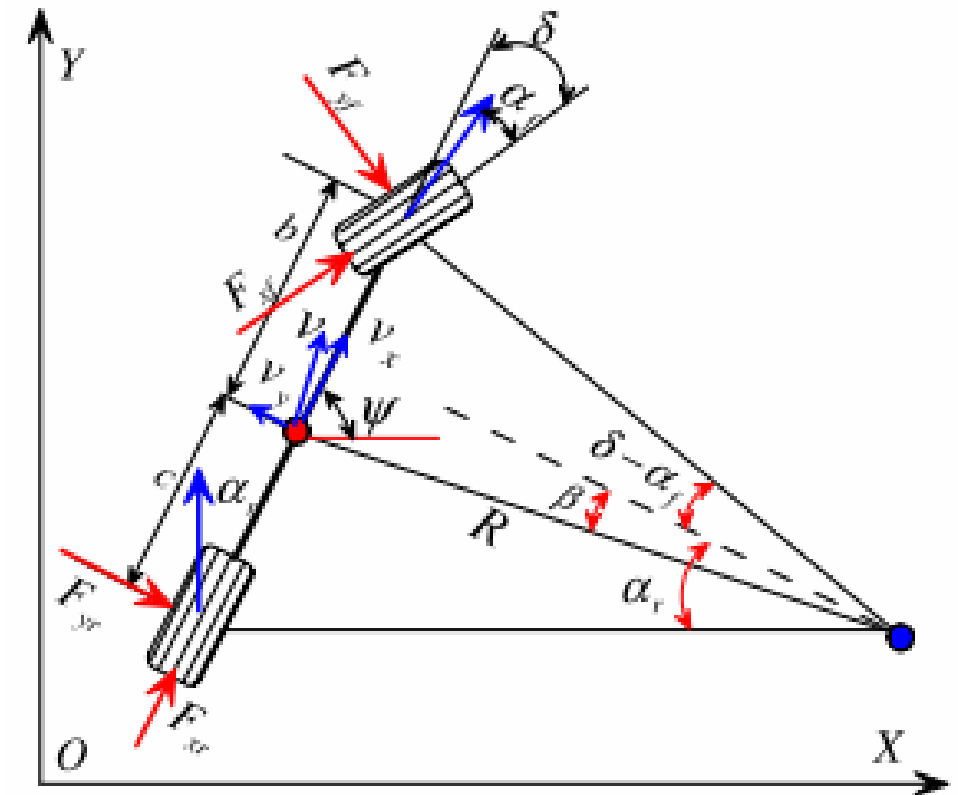


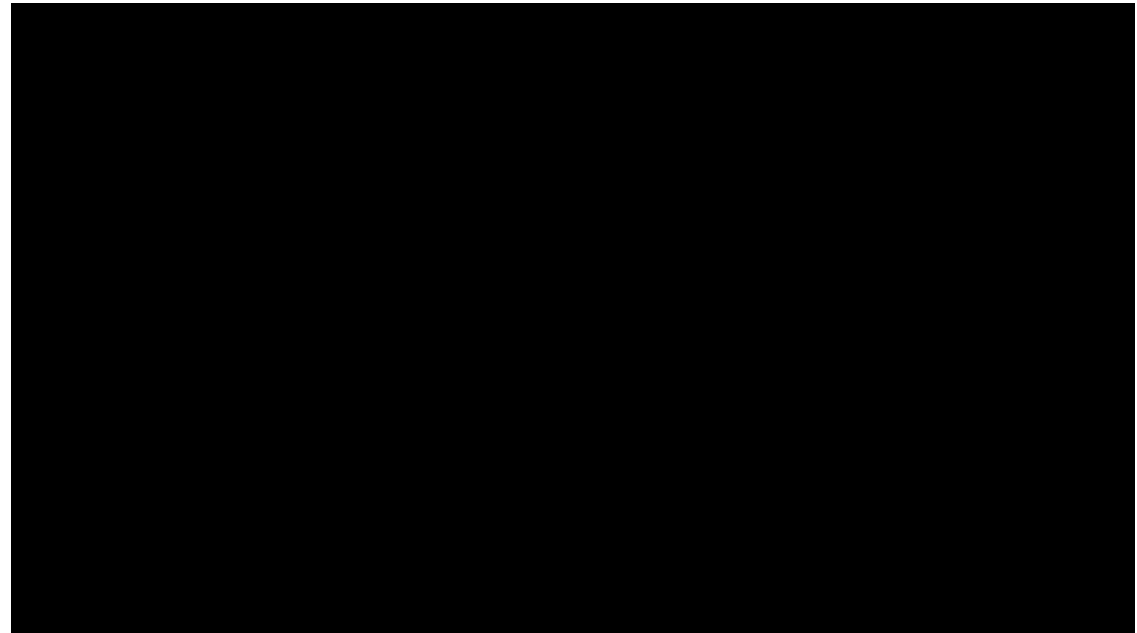
Fig. 1. Bicycle model of vehicle

Dynamic Constraints

★ Dynamic constraints

⑩ Correcting for slip

⑩ https://www.youtube.com/watch?v=itggGQu_ECc



Dynamic Constraints

★ Models increase in complexity as needed for performance tuning

⑩ Aerodynamic drag force $F_{wind} = (C_w A_w v_t^2 g) / 16$

⑩ Maximum engine torque $\frac{F_{max}}{m} = 1 + \frac{3}{1 + e^{(\frac{v_t - 12}{4})}}$

★ Each layer of dynamics:

⑩ Increases accuracy of model

⑩ Increases computational complexity

⑩ <https://youtu.be/tesD4F-HOxs?t=1m24s>

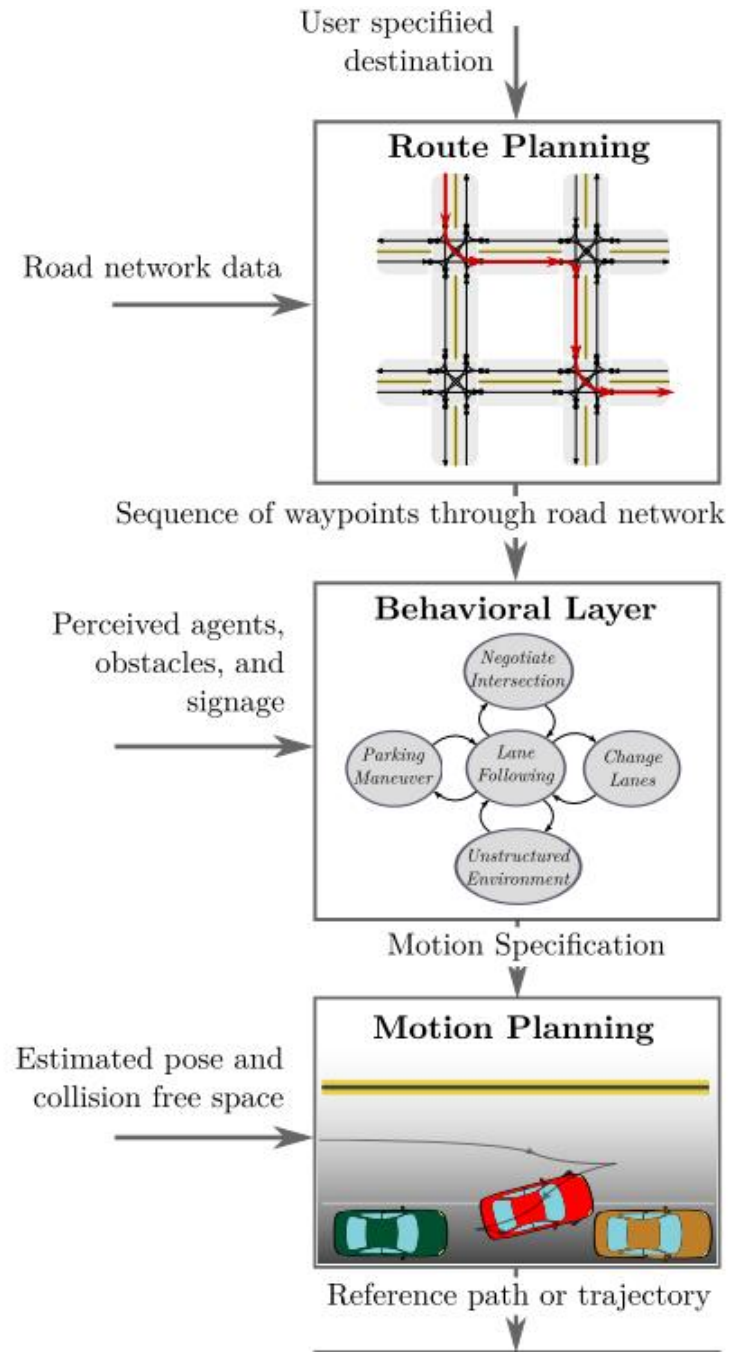


Structure

- ★ Modeling a car
- ★ Planning
 - ⑩ Route Planning
 - ⑩ Behavior Planning
 - ⑩ Motion Planning
- ★ Control
- ★ Interaction with other Drivers
- ★ Case Studies

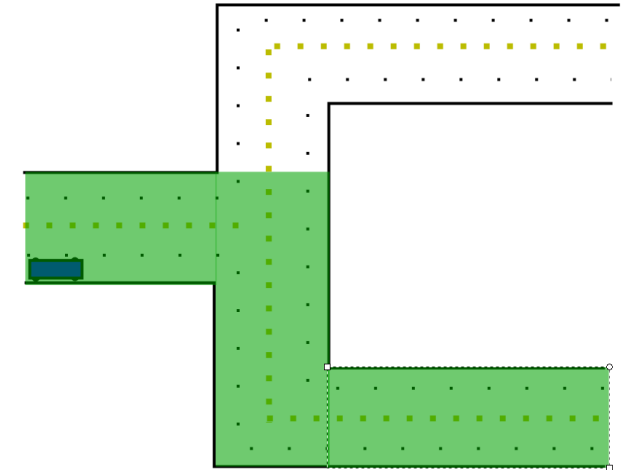
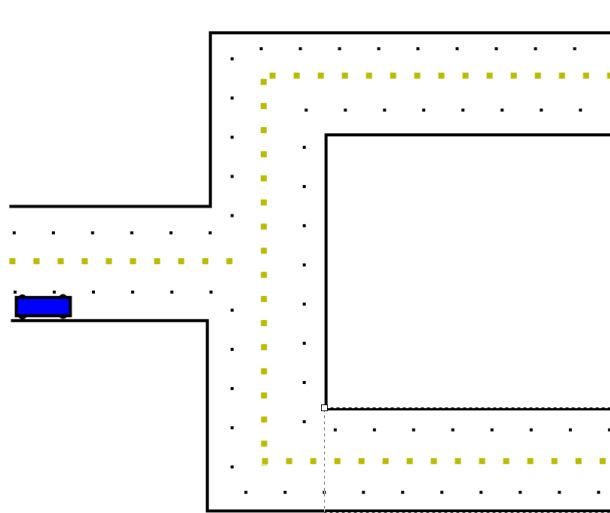
Autonomous Driving: Planning

- ✦ Route Planning
- ✦ Behavior Planning
- ✦ Motion Planning



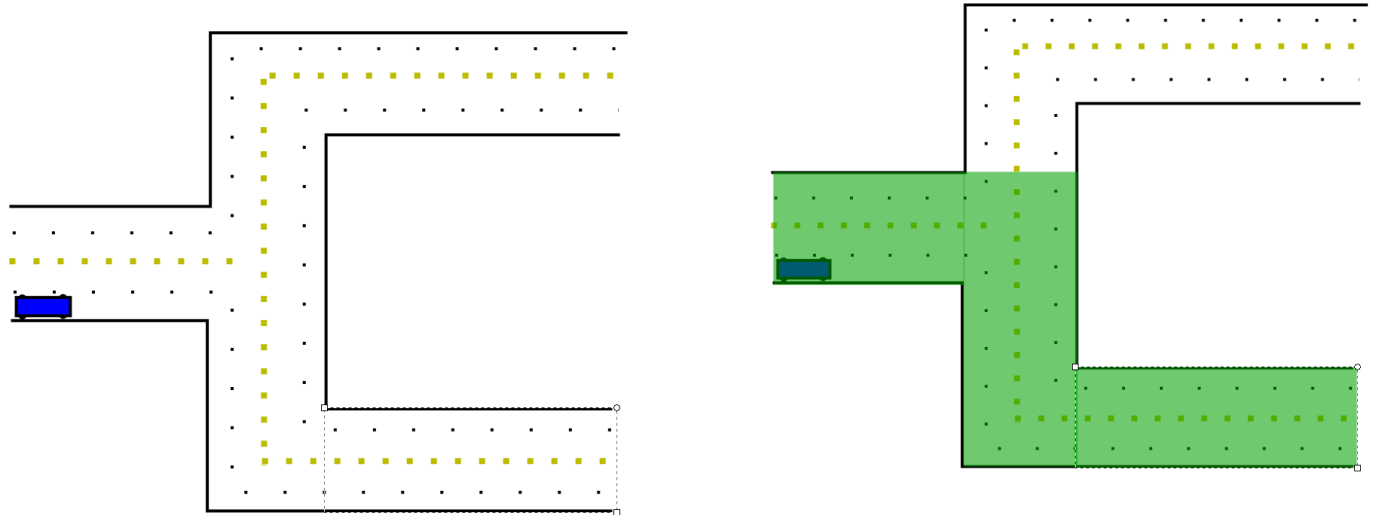
Mission Planner (Route Planning)

- ✦ Determine the appropriate macro-level route to take
- ✦ Typically road level i.e. which roads to take
- ✦ Katrakazas: “Route planning is concerned with finding the best global route from a given origin to a destination, supplemented occasionally with real-time traffic information”



Mission Planner (Route Planning)

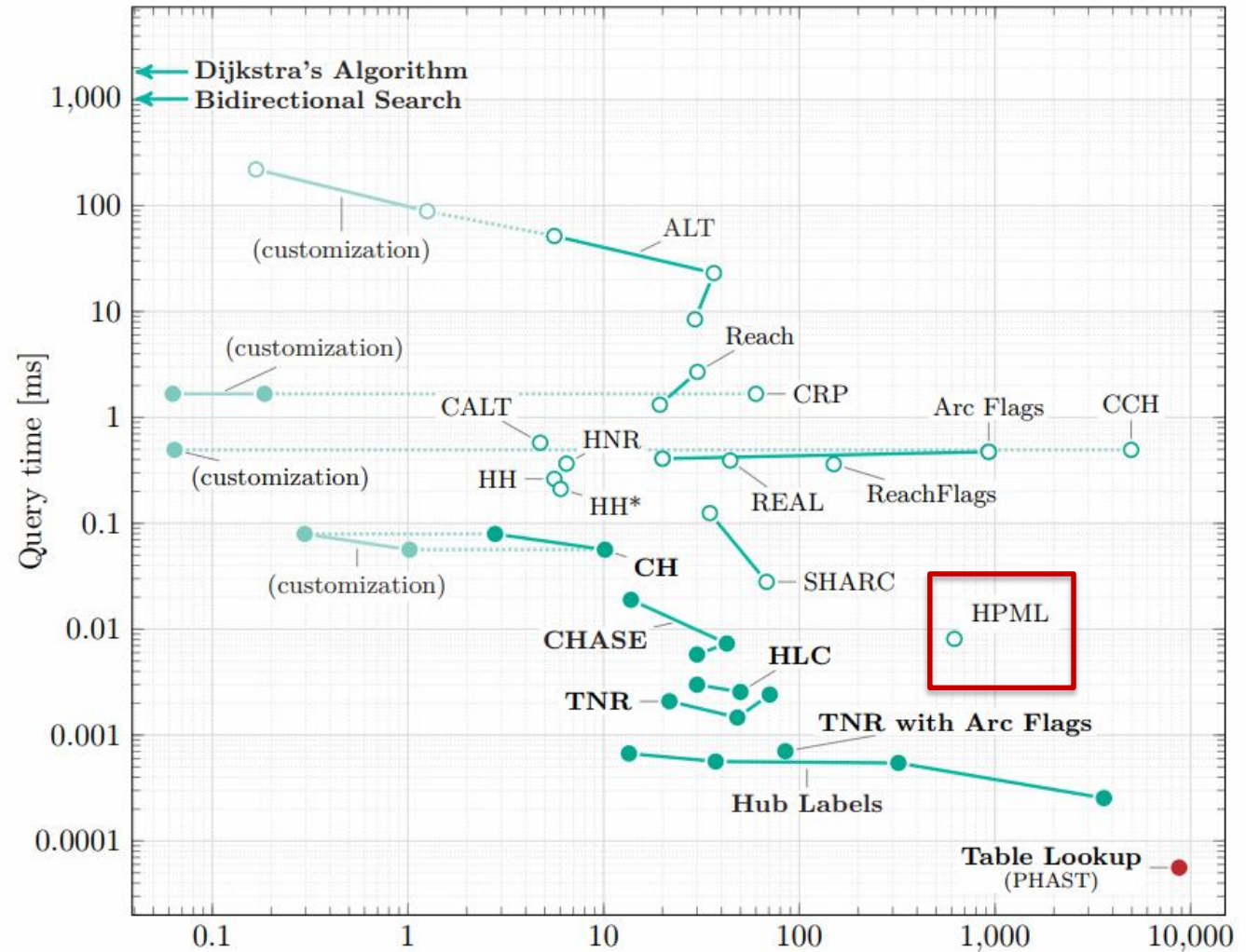
- ★ Pendleton: “considers high level objectives, such as assignment of pickup/dropoff tasks and which roads should be taken to achieve the task”
- ★ Typical approaches:
 - ⑩ RNG (Road-network Graph)
 - ★ A*
 - ★ Dijkstras
 - ⑩ Scale poorly!



Mission Planner (Route Planning)

- ★ Massive-scale algorithms needed for routing
- ★ 18 million vertices, 42.5 million edges
 - ⑩ Partial Western Europe dataset

Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., ... Werneck, R. F. (2015). Route Planning in Transportation Networks. *Microsoft Research Technical Report*, 1–65.

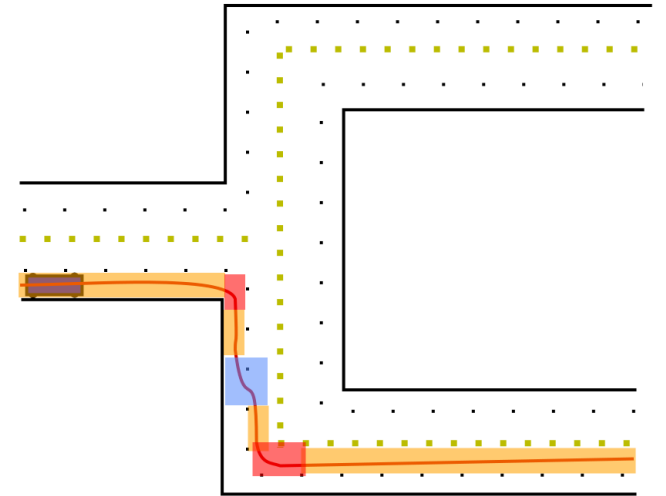
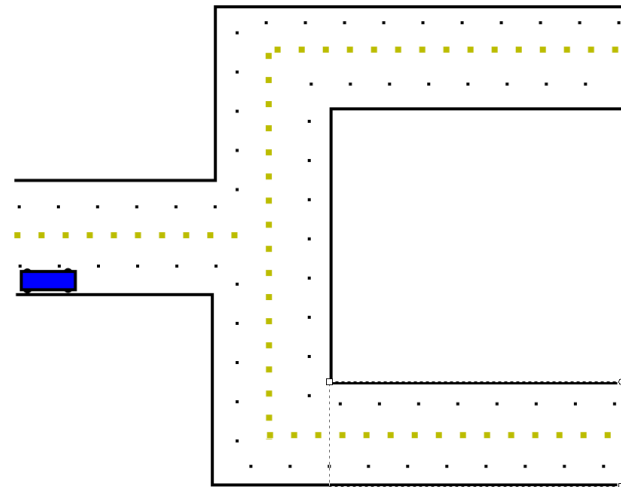


Structure

- ★ Modeling a car
- ★ Planning
 - ⑩ Route Planning
 - ⑩ **Behavior Planning**
 - ⑩ Motion Planning
- ★ Control
- ★ Interaction with other Drivers
- ★ Case Studies

Behavior Planner

- ★ “makes ad hoc decisions to properly interact with other agents and follow rules restrictions, and thereby generates local objectives, e.g., change lanes, overtake, or proceed through an intersection”
 - ⑩ Finite State Machines
 - ⑩ Finite time maneuvers



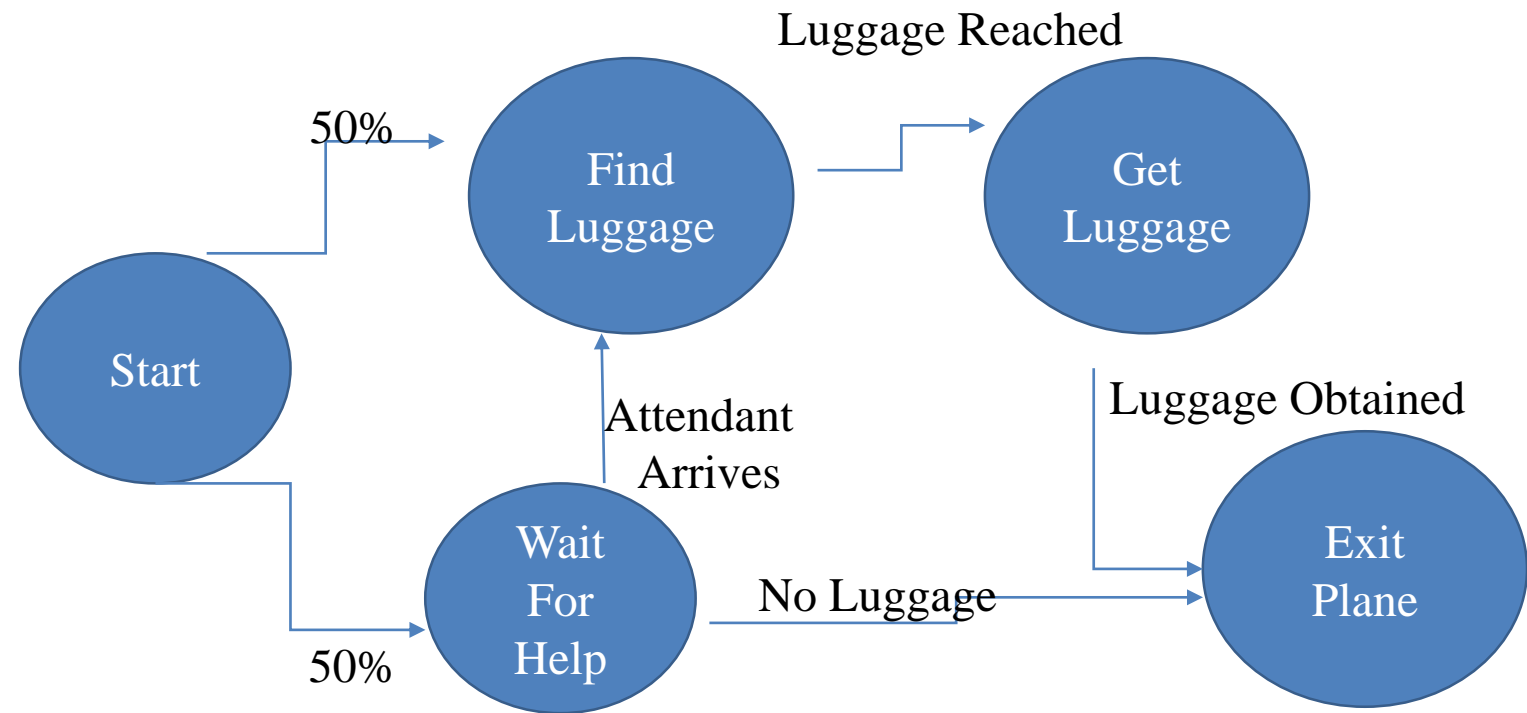
Behavior Planner

- ★ Example from crowd sim

- ★ AI Technique

- ⑩ Defines a set of States and Transition functions between them

- ⑩ Allows us to represent complex behaviors with simple components



Behavior Planner

★ Finite State Machines

- ⑩ Set of “states” and transition functions between them
- ⑩ Separate from configuration state

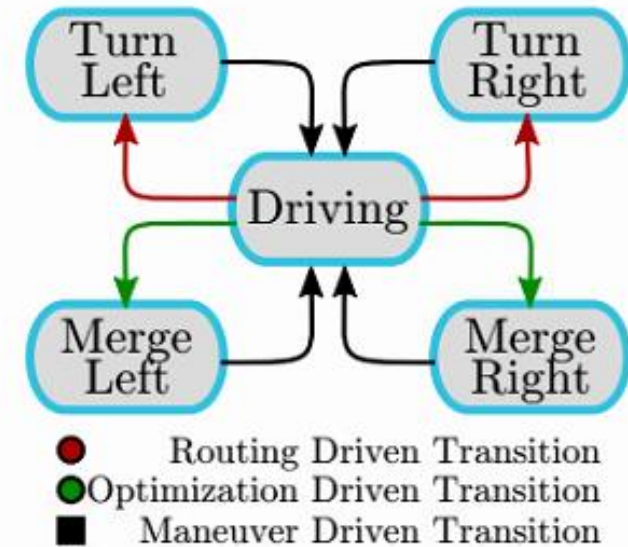
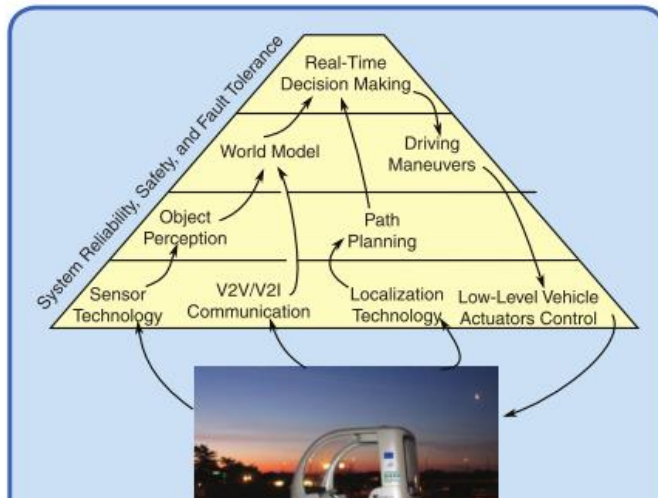


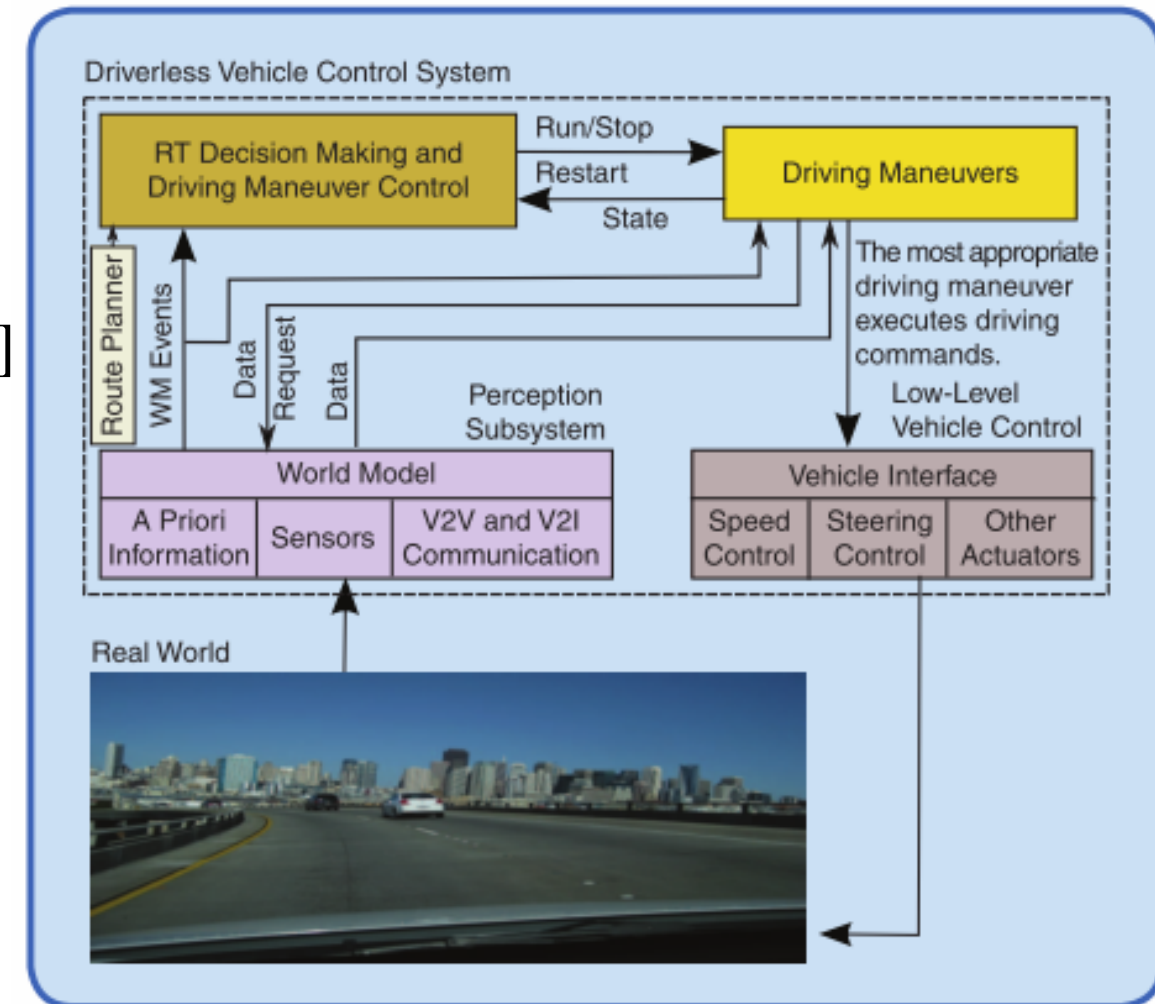
Fig. 2. **Finite State Machine:** We highlight different behavior states that are determined by the routing and optimization algorithms. When executing turns, the routing algorithm transitions the behavior state to a turning state. When the optimization-based maneuver algorithm plans a lane change, the behavior state is transitioned to merging.

Behavior Planner

- ★ FSMs limited in some cases
 - ⑩ What to do in unseen situations?
- ★ Real-time decision making [Furda et al 2011]



Furda, A., & Vlacic, L. (2011). Enabling safe autonomous driving in real-world city traffic using Multiple Criteria decision making. *IEEE Intelligent Transportation Systems Magazine*, 3(1), 4–17. <http://doi.org/10.1109/MITS.2011.940472>



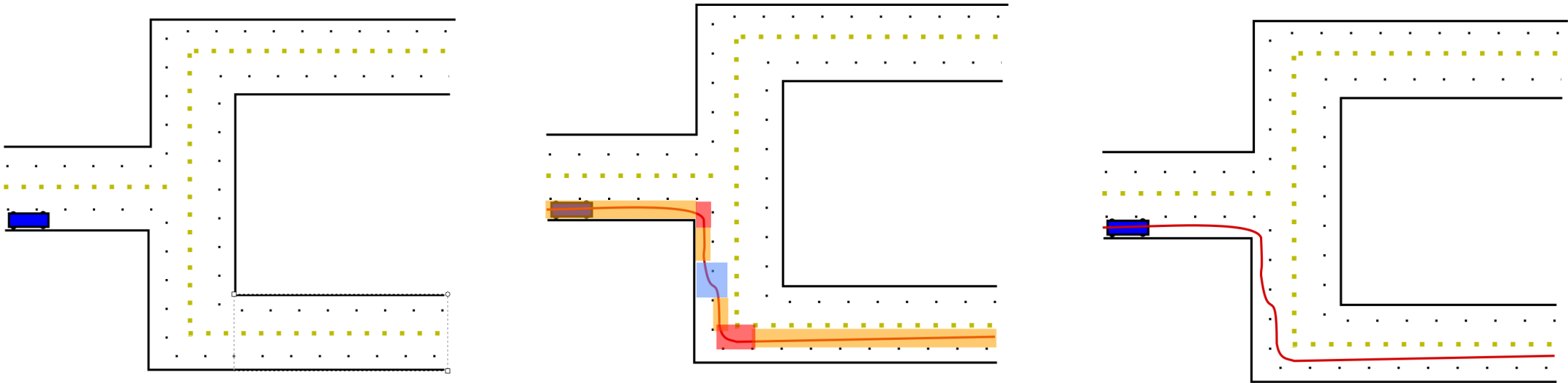
Structure

- ★ Modeling a car
- ★ Planning
 - ⑩ Route Planning
 - ⑩ Behavior Planning
 - ⑩ **Motion Planning**
- ★ Control
- ★ Interaction with other Drivers
- ★ Case Studies



Motion Planning

- ★ Pendleton: generates appropriate paths and/or sets of actions to achieve local objectives, with the most typical objective being to reach a goal region while avoiding obstacle collision



Motion Planning

★ Path Planning

⑩ A path $\sigma(\alpha): [0,1] \rightarrow \mathcal{X}$

★ Trajectory Planning

⑩ A path $\pi(t): [0,T] \rightarrow \mathcal{X}$

Problem IV.1 (Optimal path planning). Given a 5-tuple $(\mathcal{X}_{\text{free}}, \mathbf{x}_{\text{init}}, X_{\text{goal}}, D, J)$ find $\sigma^* =$

$$\begin{aligned} & \arg \min_{\sigma \in \Sigma(\mathcal{X})} J(\sigma) \\ & \text{subj. to} \quad \sigma(0) = \mathbf{x}_{\text{init}} \text{ and } \sigma(1) \in X_{\text{goal}} \\ & \quad \sigma(\alpha) \in \mathcal{X}_{\text{free}} \quad \forall \alpha \in [0, 1] \\ & \quad D(\sigma(\alpha), \sigma'(\alpha), \sigma''(\alpha), \dots) \quad \forall \alpha \in [0, 1]. \end{aligned}$$

Problem IV.2 (Optimal trajectory planning). Given a 6-tuple $(\mathcal{X}_{\text{free}}, \mathbf{x}_{\text{init}}, X_{\text{goal}}, D, J, T)$ find $\pi^* =$

$$\begin{aligned} & \arg \min_{\pi \in \Pi(\mathcal{X}, T)} J(\pi) \\ & \text{subj. to} \quad \pi(0) = \mathbf{x}_{\text{init}} \text{ and } \pi(T) \in X_{\text{goal}} \\ & \quad \pi(t) \in \mathcal{X}_{\text{free}} \quad \forall t \in [0, T] \\ & \quad D(\pi(t), \pi'(t), \pi''(t), \dots) \quad \forall t \in [0, T]. \end{aligned}$$



Motion Planning

★ How do we evaluate them?

⑩ Complexity (computation cost)

★ limits how frequently we can replan

★ NEVER get it perfectly right, so we focus on replanning as fast as possible

⑩ Completeness (likelihood that a solution will be found if one exists)

★ The piano-movers problem is PSPACE-HARD



Motion Planner

★ Basic overview

- ⑩ Complete planning
- ⑩ Combinatorial Planning
- ⑩ Sample-Based planning



Motion Planner

★ Basic overview

- ⑩ **Complete planning** - continuous plan in configuration space
 - ★ Exponential in dimensions of c-space (curse of dimensionality)
 - ★ "Complete"
- ⑩ Combinatorial Planning - discrete planning over an exact decomposition of the configuration space
- ⑩ Sample-Based planning:



Motion Planner

- ★ Basic overview

- ⑩ Complete planning

- ⑩ **Combinatorial Planning** - discrete planning over an exact decomposition of the configuration space

- ★ Exponential in dimensions of c-space discretization (curse of dimensionality)

- ★ "resolution complete"

- ⑩ Sample-Based planning



Motion Planner

- ★ Basic overview

- ⑩ Complete planning

- ⑩ Combinatorial Planning

- ⑩ **Sample-Based planning** - Sample in space to find controls / positions which are collision free and linked

- ★ Probabilistically complete

- ⑩ Some “probabilistically optimal”

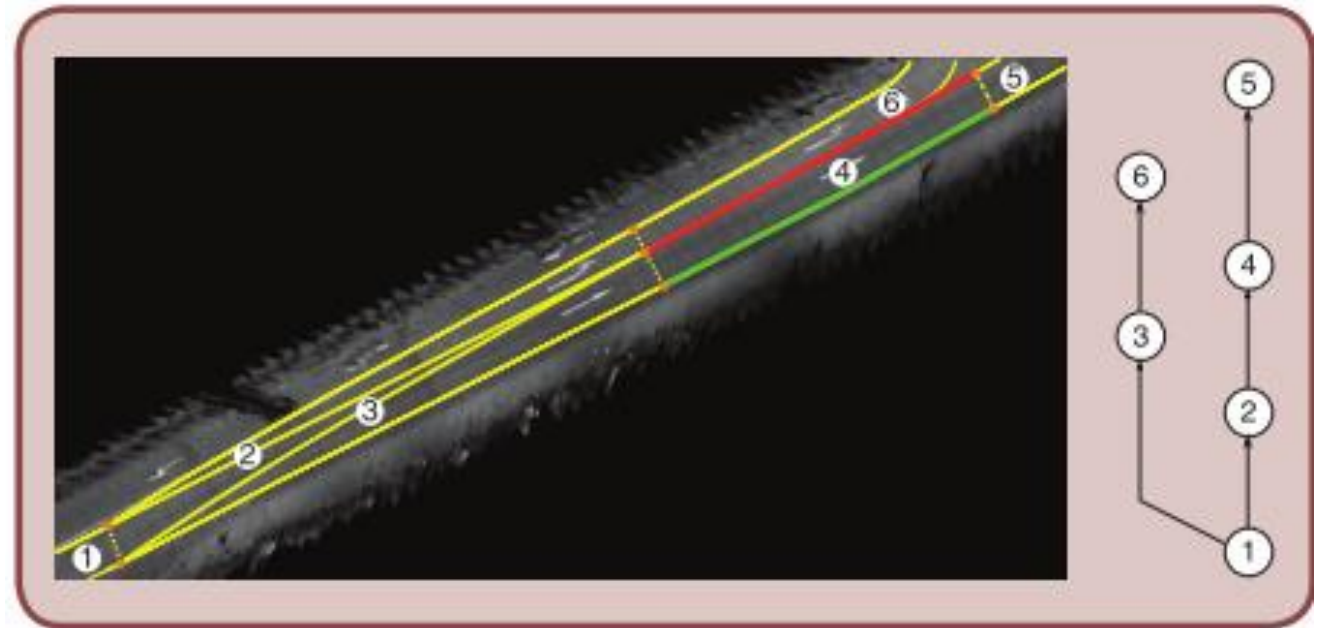
- ★ NOT exponential in configuration space



Motion Planner: Combinatorial Planners

★ Driving Corridors:

- ⑩ Decompose lanes into polygonal lanelets
- ⑩ Represent obstacles as polygonal bounding boxes or overlapping discs
- ⑩ Adjust lanelets to obstacle constraints



Ziegler, J., Bender, P., Schreiber, M., Lategahn, H., Strauss, T., Stiller, C., ...
Zeeb, E. (2014). Making bertha drive-an autonomous journey on a historic route.
IEEE Intelligent Transportation Systems Magazine, 6(2), 8–20.
<http://doi.org/10.1109/MITS.2014.2306552>



Motion Planner: Combinatorial Planners

★ Driving Corridors:

- ⑩ Decompose lanes into polygonal lanelets
- ⑩ Represent obstacles as polygonal bounding boxes or overlapping discs
- ⑩ Adjust lanelets to obstacle constraints

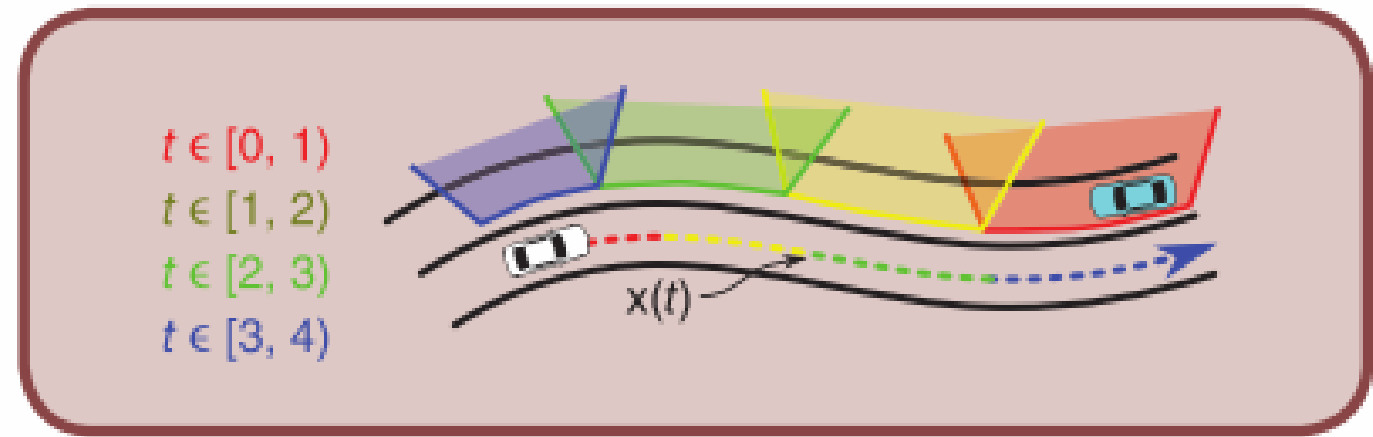


FIG 10 Constraints for an oncoming Object (cyan). The trajectory is only constrained by polygons of corresponding color.

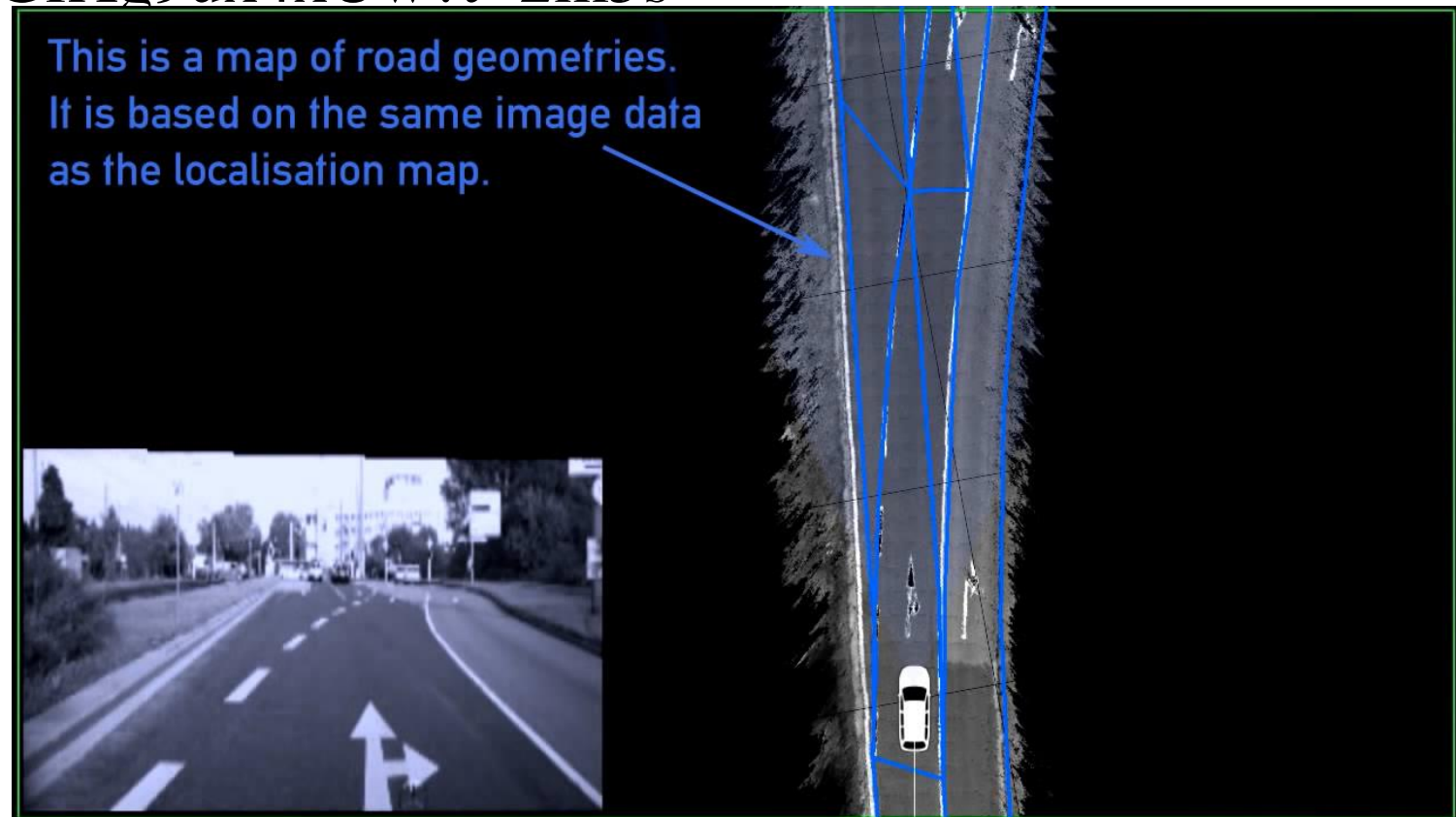
Ziegler, J., Bender, P., Schreiber, M., Lategahn, H., Strauss, T., Stiller, C., ...
Zeeb, E. (2014). Making bertha drive-an autonomous journey on a historic route.
IEEE Intelligent Transportation Systems Magazine, 6(2), 8–20.
<http://doi.org/10.1109/MITS.2014.2306552>



Motion Planner: Combinatorial Planners

★ Driving Corridors:

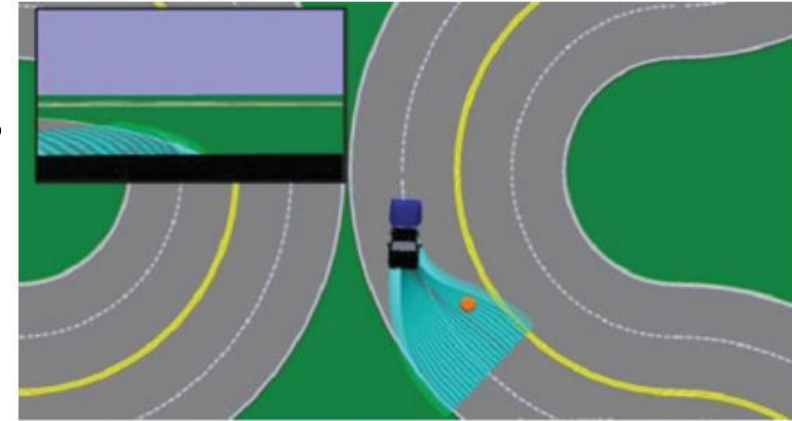
⑩ <https://youtu.be/GfXg9ux4xUw?t=2m5s>



Motion Planner: Combinatorial Planners

★ Darpa Urban Challenge:

- ⑩ BOSS: kinodynamic reachable set
- ⑩ Trajectory planner generates candidate trajectories and goals
 - ★ Done by precomputation of many curves
- ⑩ “best” trajectory chosen by optimization



Urmson, C., Baker, C., Dolan, J., Rybski, P., Salesky, B., Whittaker, W., ... Darms, M. (2009). Autonomous Driving in Traffic: Boss and the Urban Challenge. *AI Magazine*, 30(2), 17–28. <http://doi.org/10.1002/rob>



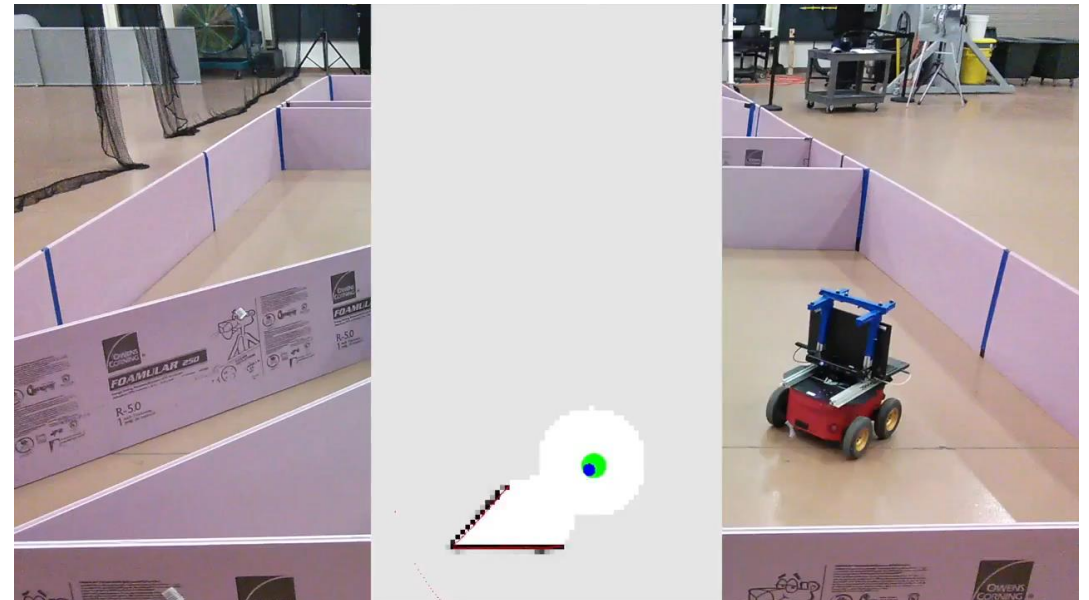
Motion Planner: Combinatorial Planners

★ Grid Decomposition approaches:

- ⑩ Generate cellular-grid representation of local space
- ⑩ Cells encode probability of occupancy
- ⑩ Moving obstacles propagate occupancy probability

★ Grid Decomposition approaches:

- ⑩ <https://youtu.be/CRQfhhICSj0>
- ⑩ <https://youtu.be/MzpBzrtEGrA>



Motion Planner: Sample-based Planners

- ★ Pendleton: popular for their guarantees of probabilistic completeness, that is to say that given sufficient time to check an infinite number of samples, the probability that a solution will be found if it exists converges to one.
- ★ General approaches:
 - ⑩ PRM: Probabilistic Roadmaps
 - ⑩ RRT: Rapidly-Exploring Random Tree
 - ⑩ FMT: Fast-Marching Trees



Motion Planner: Sample-based Planners

★ PRM: Probabilistic Roadmaps $G(V, E)$

⑩ Repeat until n collision-free samples found i.e. $|V| = n$

★ Sample a point p in configuration space

★ $V = V \cup p$ if p is collision-free

⑩ For each vertex $v_i \in V$

★ Find k nearest neighbors: $N = \{v_{\{i1\}}, v_{\{i2\}}, \dots, v_{\{ik\}}\}$

★ For each neighbor $v_{ij} \in N$

⑩ Determine if edge $e = (v_i, v_{ij})$ is collision-free

⌚ $E = E \cup e$



Maneuver Planner: Sample-based Planners

★ RRT: Rapidly-Exploring Random Tree

⑩ Given at-least one initial configuration in free-space and a goal configuration

- ★ Sample a point p in configuration space, determine if it is collision free
- ★ If so, find nearest node n to the point, move some δ towards the point
- ★ If n to $n + \delta$ is CLEAR, connect to the tree



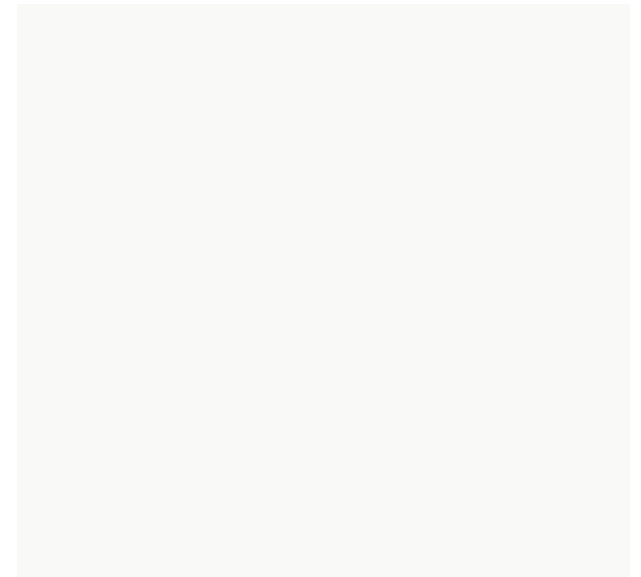
Motion Planner: Sample-based Planners

★ RRT:

⑩ <https://www.youtube.com/watch?v=rPgZyq15Z-Q>

⑩ <https://www.youtube.com/watch?v=mEAr2FBUJEI>

⑩ <https://www.youtube.com/watch?v=p3p0EWT5lpw>



Motion Planner: Sample-based Planners

- ★ Sample-based Planning specifically for cars:
 - ⑩ Dynamics computation
 - ⑩ Complex state evolution
- ★ Pendleton: “Incorporating differential constraints into state-sampling planners is still a challenging matter, and requires a steering function to draw an optimal path between two given states which obeys control constraints (if such a path exists), as well as efficient querying methods to tell whether a sampled state is reachable from a potential parent state”



Maneuver Planner: Sample-based Planners

★ PRM with dynamics

- ⑩ Extends existing PRM framework
- ⑩ Sampling directly from admissible controls
- ⑩ State \times time space formulation
- ⑩ Set of differential equations describing all possible local motions of a robot

Hsu, D., Kindel, R., Latombe, J.-C., & Rock, S. (2002). Randomized Kinodynamic Motion Planning with Moving Obstacles. *The International Journal of Robotics Research*, 21(3), 233–255. <http://doi.org/10.1177/027836402320556421>



Maneuver Planner: Sample-based Planners

Algorithm 1 Control-driven randomized expansion.

1. Insert m_b into T ; $i \leftarrow 1$.
 2. **repeat**
 3. Pick a milestone m from T with probability $\pi_T(m)$.
 4. Pick a control function u from \mathcal{U}_ℓ uniformly at random.
 5. $m' \leftarrow \text{PROPAGATE}(m, u)$.
 6. **if** $m' \neq \text{nil}$ **then**
 7. Add m' to T ; $i \leftarrow i + 1$.
 8. Create an edge e from m to m' ; store u with e .
 9. **if** $m' \in \text{ENDGAME}$ **then** exit with SUCCESS.
 10. **if** $i = N$ **then** exit with FAILURE.
-

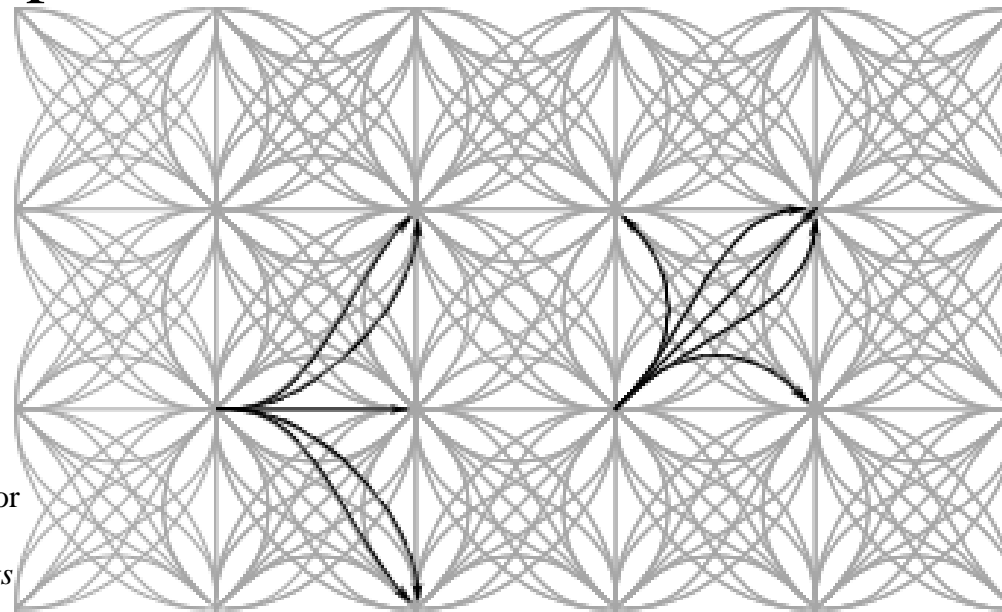


Maneuver Planner: Sample-based Planners

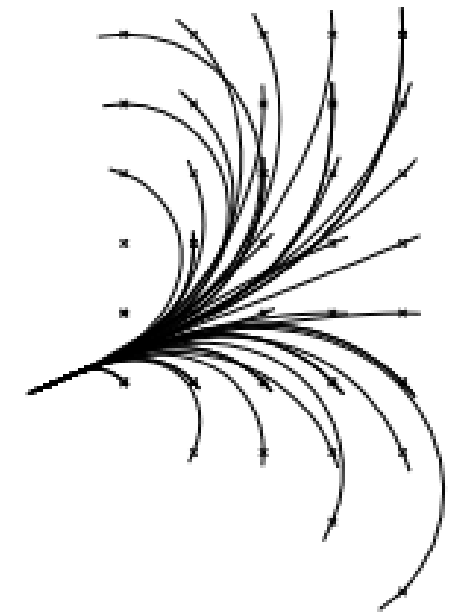
★ State-lattice planners

⑩ Ex: Configurations in space

Ziegler, J., & Stiller, C. (2009). Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios. *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, 1879–1884.
<http://doi.org/10.1109/IROS.2009.5354448>



(a)



(b)

Maneuver Planner: Sample-based Planners

★ State-lattice planners

⑩ <https://www.youtube.com/watch?v=I5hL8vSo6DI>

⑩ Notice the discrete maneuver points

ON-ROAD STATE LATTICE
COST FUNCTION COMPARISON

MATTHEW MCNAUGHTON



Structure

- ★ Modeling a car
- ★ Planning
- ★ **Control**
 - ⑩ PID
 - ⑩ Model Predictive Control
 - ⑩ Path/Trajectory tracking
- ★ Interaction with other Drivers
- ★ Case Studies

Autonomous Driving: Control

★ Control

- ⑩ Executing the planned maneuvers accounting for error / uncertainty
- ⑩ Commands sent to actuators

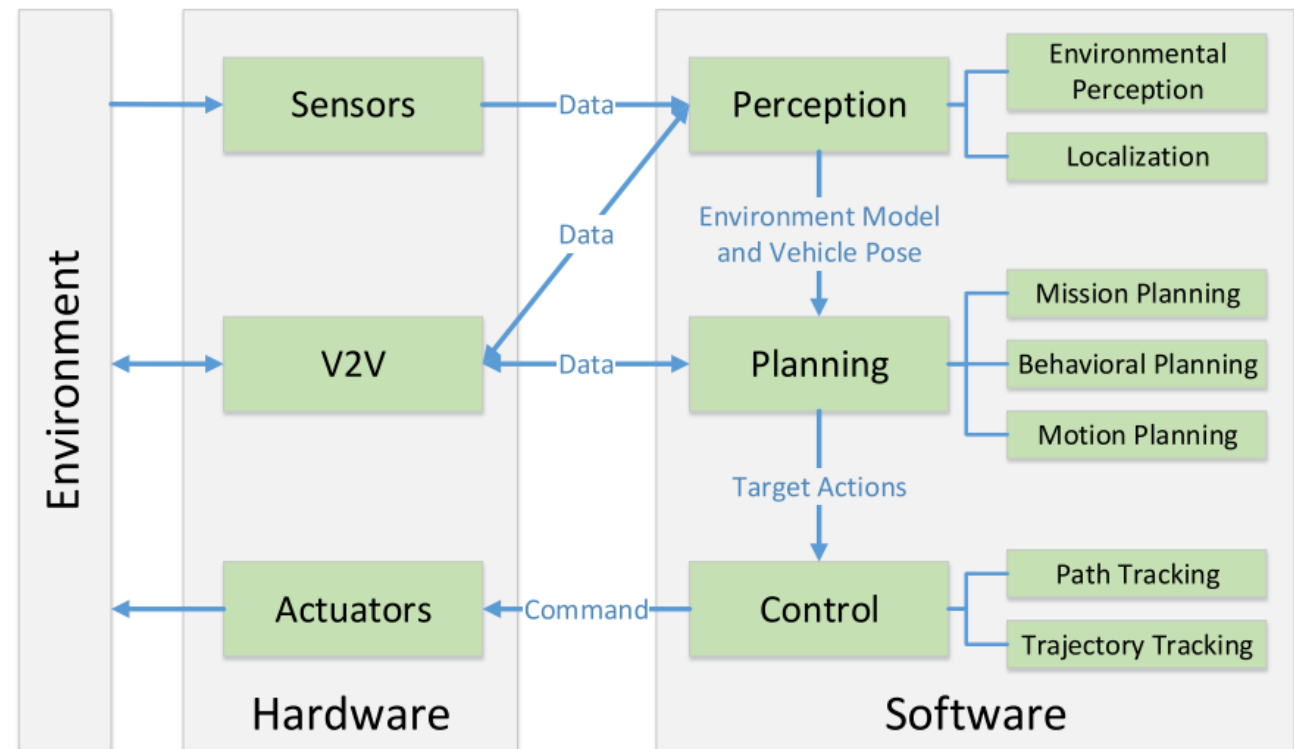
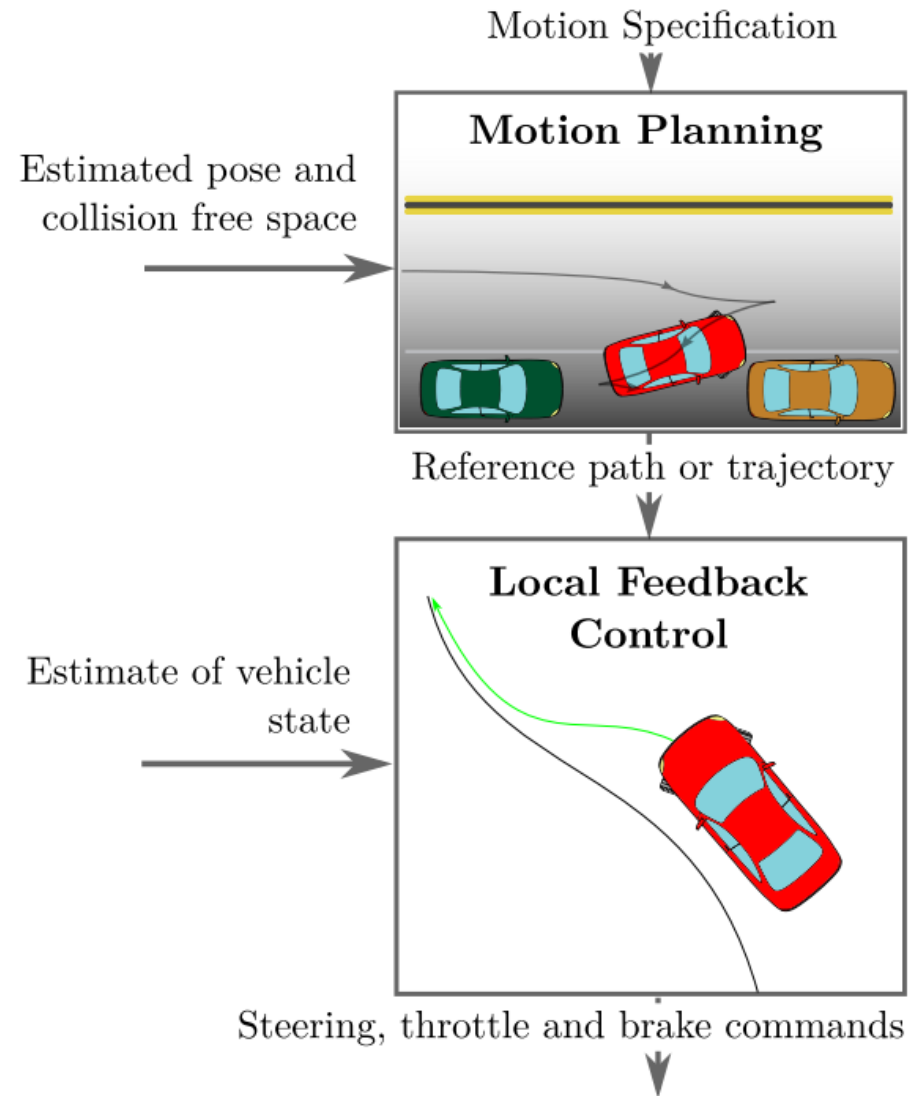


Figure 2. A typical autonomous vehicle system overview, highlighting core competencies.



Autonomous Driving: Control



Control: Core Concepts

★ Open-loop control examples

⑩ Timers:

- ★ Electronic timing switches
- ★ Clothes Dryer

⑩ Simple throttle (non-electronic)

- ★ Motorbikes, go-karts
- ★ Stove-top gas

⑩ Sinks / simple valves

- ★ Hot water / cold water



Control: Core Concepts

★ Closed-loop control examples

⑩ Thermostat:

- ★ Engages air-conditioning depending on temperature

⑩ Oven:

- ★ Heating element controlled by temperature

⑩ Cruise-control:

- ★ Throttle controlled by current speed / acceleration

⑩ Used EXTENSIVELY in plant control (i.e. chemical, energy)



Control: Core Concepts

- ✦ Process Variable (PV): The system output we wish to control
- ✦ Set Point (SP): Target value of the process Variable
- ✦ Control Output (CO): Output of the controller (input to the system)
- ✦ Error (E): Difference between SP and PV

<https://www.dataforth.com/introduction-to-pid-control.aspx>



Control: Core Concepts

★ Example: Water Plant Thermal Control

⑩ Water kept at constant temperature by gas heater

⑩ If level rises, gas reduced to stabilize

★ PV: Temperature of water

★ SP: Desired Temperature

★ CO: Level of gas applied to burner

<https://www.dataforth.com/introduction-to-pid-control.aspx>

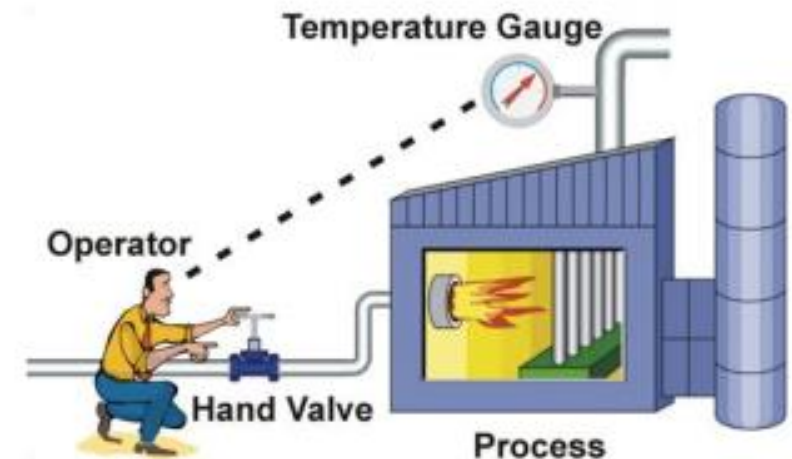


Figure 1
An Operator Performing Manual Control

Control: Core Concepts

✦ Can we replace the manual control with automatic controller?

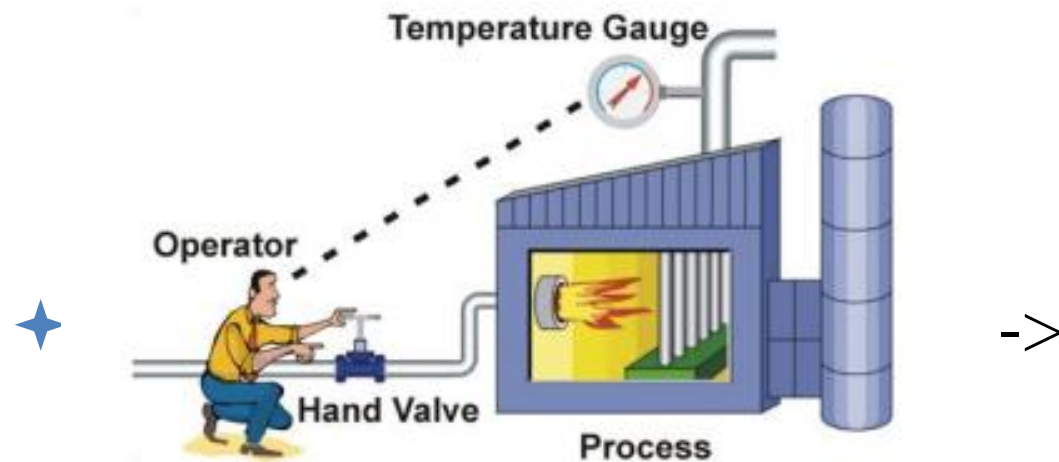


Figure 1
An Operator Performing Manual Control

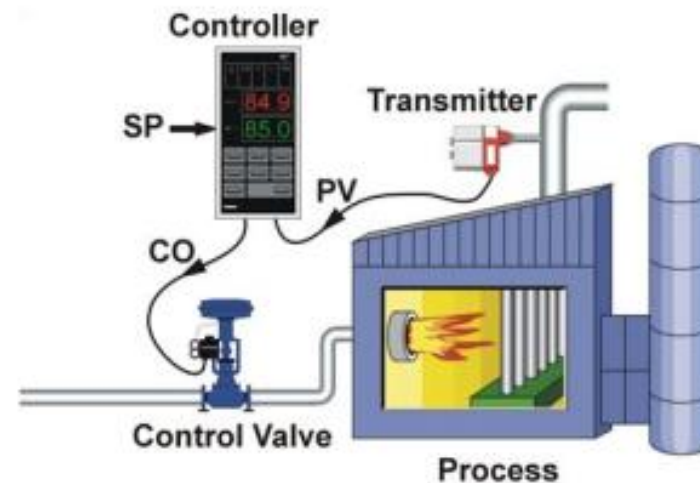


Figure 2
A PID Controller Performing Automatic Control

✦ Of course, we can!

Structure

- ✦ Modeling a car
- ✦ Planning
- ✦ **Control**
 - ⑩ **PID**
 - ⑩ Model Predictive Control
 - ⑩ Path/Trajectory tracking
- ✦ Interaction with other Drivers
- ✦ Case Studies



Control: PID

- ★ **Proportional-Integral-Derivative** Controller: control loop feedback mechanism widely used in industrial control systems and a variety of other applications requiring continuously modulated control.
 - ⑩ Continuously calculates E, applies correction based on proportional, integral, and derivative terms (denoted P, I, and D respectively)
 - ⑩ Proportion (P): Current error, E (typically $SP - PV$)
 - ⑩ Integral (I): integral of E (sum of errors over time)
 - ⑩ Derivative (D): derivative of E (typically finite difference)



Control: PID

- ★ **Proportional-Integral-Derivative** Controller: control loop feedback mechanism widely used in industrial control systems and a variety of other applications requiring continuously modulated control.

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$



Control: PID

- ✦ Proportion: Output controlled by error and Controller Gain (K_p)
- ✦ Control output proportional to error
 - ⑩ Choice of error function, but typically $SP - PV$
- ✦ High gain: can cause oscillation
- ✦ Low gain: fails to correct to Set Point

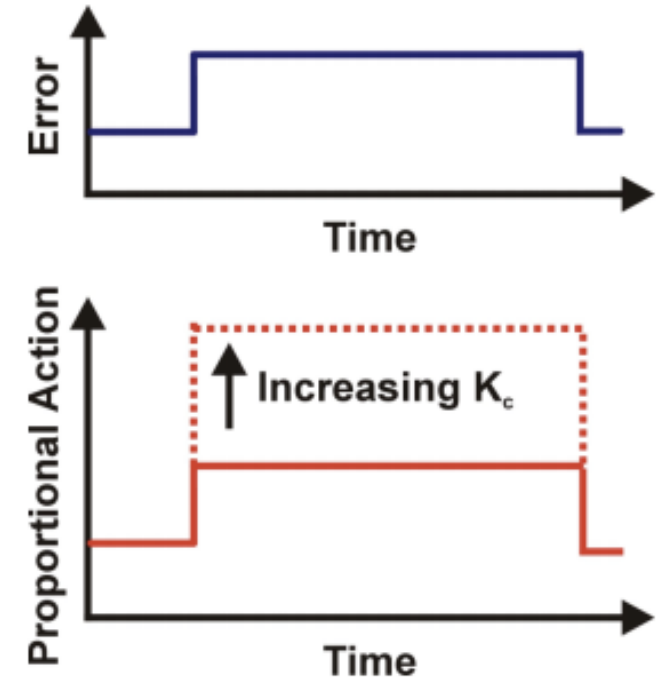


Figure 3
Proportional Control Action

Control: PID

- ✦ Proportion-only controller: Output controlled by error and Controller Gain (K_p)
- ✦ Control output proportional to error
 - ⑩ Choice of error function, but typically $SP - PV$
- ✦ Add bias point for steady output at 0 error

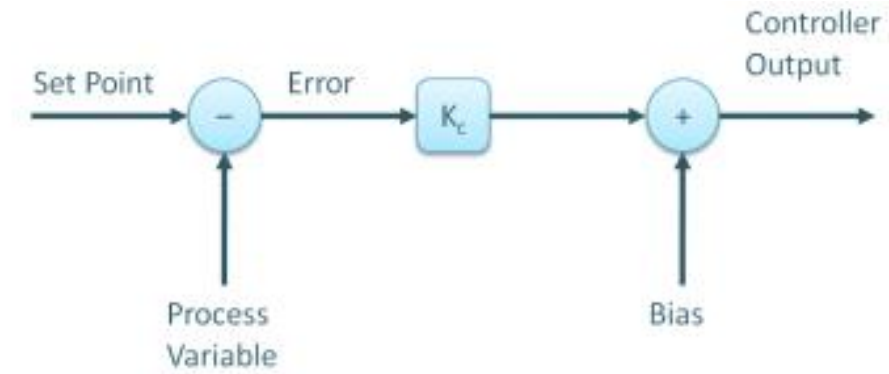


Figure 4
A Proportional-Only Controller Algorithm

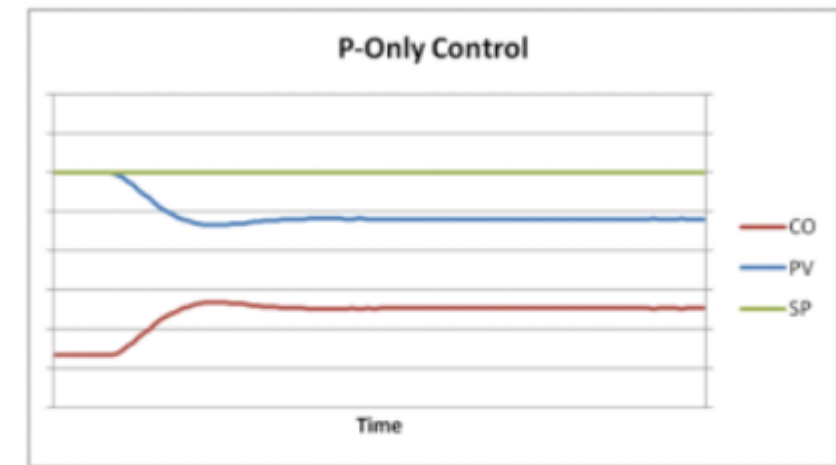


Figure 6
A Proportional Controller's Response to a Disturbance



Control: PID

- ✦ Integral Control: Output term controlled by integral of error and Integral Gain (K_i)
- ✦ Corrects “steady-state” error
- ✦ Requires a “time” factor for integration (T_i)
- ✦ Longer time = less integral action

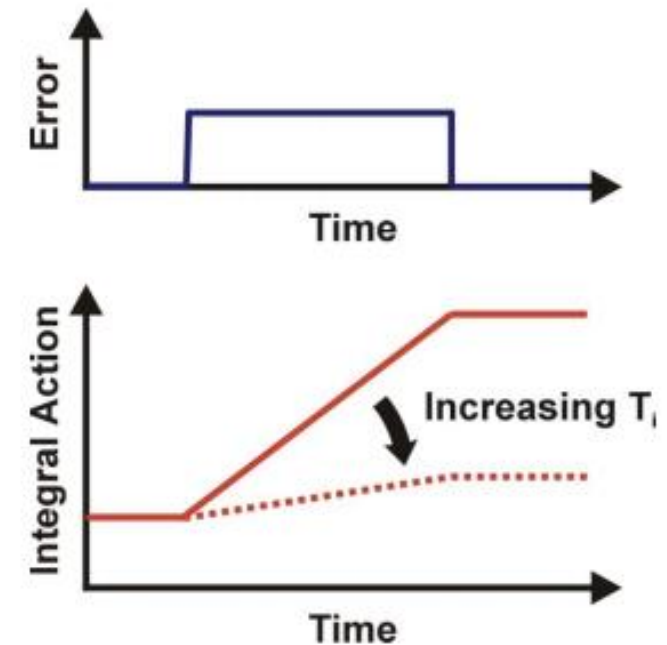


Figure 7
Integral Control Action

Control: PID

- ★ PI Controller: Proportion and integral terms
- ★ Corrects steady-state error, converges rather than oscillates

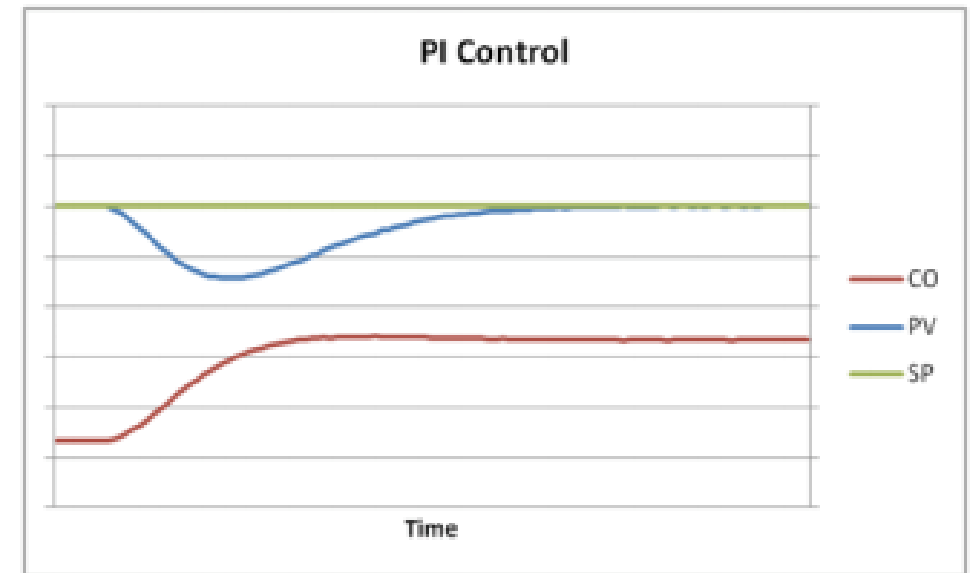


Figure 9
A PI Controller's Response to a Disturbance



Control: PID

- ✦ Derivative: Output term controlled by derivative of error and Derivative Gain (K_d)
- ✦ Assists in rapid response to disturbance
- ✦ Requires time parameter to operate

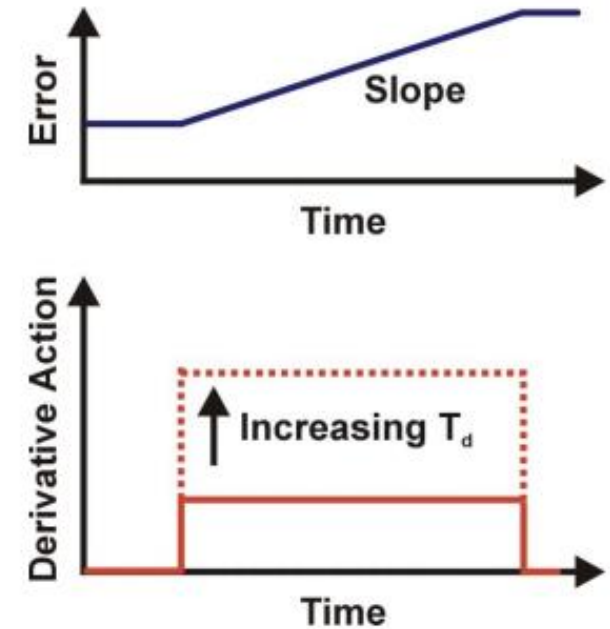


Figure 10
Derivative Control Action

Control: PID

- ★ PID Controller: Proportion, Integral, Derivative terms
- ★ Complete closed-loop controller
 - ⑩ Used in AutonoVi and countless applications

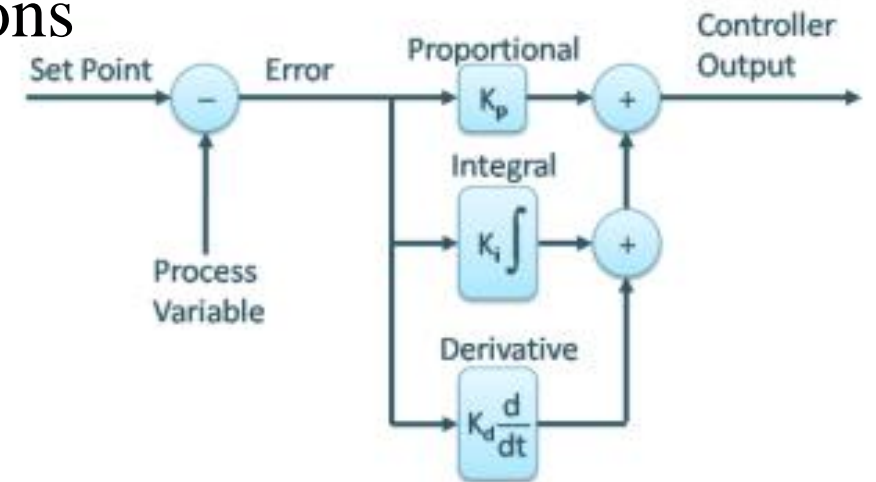


Figure 12
The Parallel PID Controller Algorithm



Control: PID Tuning

★ Rules of thumb for tuning a PID controller:

★ https://upload.wikimedia.org/wikipedia/commons/3/33/PID_Compensation_Animated.gif

Effects of increasing a parameter independently^{[20][21]}

Parameter	Rise time	Overshoot	Settling time	Steady-state error	Stability
K_p	Decrease	Increase	Small change	Decrease	Degrade
K_i	Decrease	Increase	Increase	Eliminate	Degrade
K_d	Minor change	Decrease	Decrease	No effect in theory	Improve if K_d small



Control: PID Tuning

★ Ziegler–Nichols Tuning

⑩ Tune K_p until the control loop begins to oscillate

★ Called Ultimate control point (K_u)

⑩ K_u and oscillation period T_u used to tune parameters as follows

Ziegler–Nichols method

Control Type	K_p	K_i	K_d
<i>P</i>	$0.50K_u$	—	—
<i>PI</i>	$0.45K_u$	$0.54K_u/T_u$	—
<i>PID</i>	$0.60K_u$	$1.2K_u/T_u$	$3K_uT_u/40$



Control: PID Examples

★ More examples of PID:

- ⑩ Cruise-control

- ⑩ Quad-rotor Autopilot

- ⑩ Mobile robot control

 - ★ PID for steering + PID for speed

- ⑩ Spaceships

- ⑩ ...

- ⑩ ...

- ⑩ Innumerable examples of PID control



Structure

- ★ Modeling a car
- ★ Planning
- ★ **Control**
 - ⑩ PID
 - ⑩ **MPC**
 - ⑩ Path Tracking
- ★ Interaction with other Drivers
- ★ Case Studies

Control: MPC

- ★ **Model-Predictive** Controller: control loop relying on an underlying system model to generate feed-forward control
 - ⑩ At each time step, compute control by solving an openloop optimization problem for the prediction horizon
 - ⑩ Apply the first value of the computed control sequence
 - ⑩ At the next time step, get the system state and re-compute
- ★ Resources
 - ⑩ <https://www.youtube.com/watch?v=oMUtYZOgsng>
 - ⑩ <https://www.youtube.com/watch?v=DFqOf5wbQtc>



Control: MPC

- ★ MPC is very useful when process model is available
 - ⑩ Reduces overshoot substantially
 - ⑩ Using cached table of input responses, optimization can be done quickly
- ★ MPC uses in automotive context:
 - ⑩ Traction control [Borelli 2006]
 - ⑩ Braking control [Falcone 2007]
 - ⑩ Steering [Falcone 2007]
 - ⑩ Lane-keeping [Liu 2015]



Structure

- ★ Modeling a car
- ★ Planning
- ★ Control
 - ⑩ PID
 - ⑩ MPC
 - ⑩ Path Tracking
- ★ Interaction with other Drivers
- ★ Case Studies

Control: Path tracking with controllers

- ★ Given a path/trajectory computed by the motion planner, we use controls to follow or “achieve” the path
- ★ Many methods for path/trajectory tracking:
 - ⑩ Pure-pursuit
 - ⑩ AutonoVi (Arcs)
 - ⑩ Kinematic Bicycle
 - ⑩ Model-Predictive Control



Control: Path tracking with controllers

★ Pure-pursuit

⑩ Given a geometric path, track a point ahead of the vehicle according to a fixed lookahead (can be a function of speed)

⑩ <https://www.youtube.com/watch?v=qG70QJJ8Qz8>

⑩ <https://www.youtube.com/watch?v=vlyTthJugRQ>

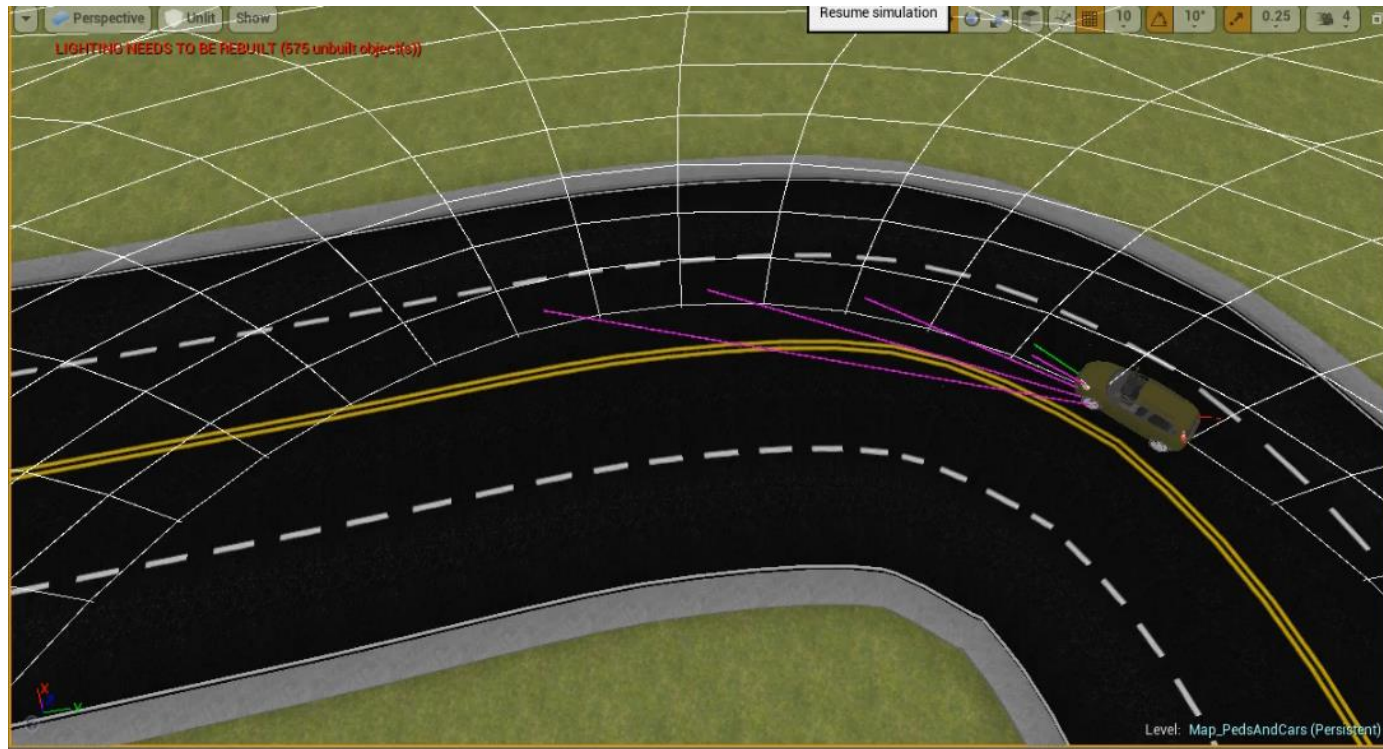
★ Advantages: simple, robust to perturbation

★ Disadvantages: Corner-cutting, oscillation for non-holonomic robots



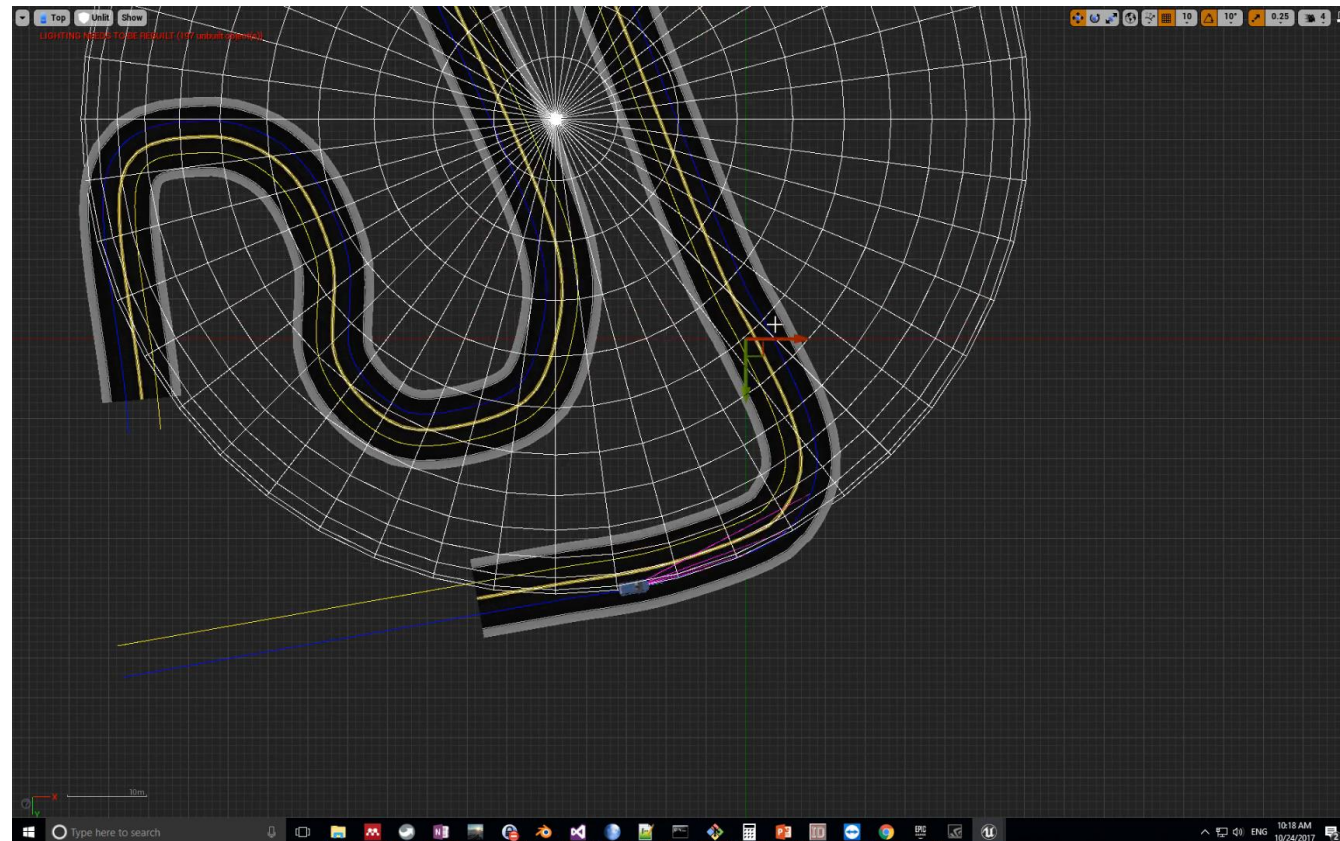
Control: Path tracking with controllers

★ AutonoVi



Control: Path tracking with controllers

★ AutonoVi



Structure

- ★ Modeling a car
- ★ Planning
- ★ Control
- ★ Interaction with other drivers
 - ⑩ Formal framework for 2-way interactions
 - ⑩ Probabilistic reasoning for multi-vehicle interactions
- ★ Case Studies



Formal Framework for 2-way interactions

- ★ Key insight is that other drivers do not operate in isolation:
 - ⑩ an autonomous car's actions will actually have effects on what other drivers will do.
 - ⑩ Leveraging these effects during planning will generate behaviors for autonomous cars that are more efficient and communicative.
- ★ Dorsa Sadigh, Shankar Sastry, Sanjit A. Seshia, and Anca D. Dragan. **Planning for Autonomous Cars that Leverages Effects on Human Actions.** In Proceedings of the Robotics: Science and Systems Conference (RSS), June 2016.



Structure

- ★ Modeling a car
- ★ Planning
- ★ Control
- ★ Interaction with other drivers
 - ⑩ Formal framework for 2-way interactions
 - ⑩ Probabilistic reasoning for multi-vehicle interactions
- ★ Case Studies



Behavior Prediction

- ★ Choose ego-vehicle actions that maximize a reward function over time within a dynamic, uncertain environment with tightly coupled inter-actions between multiple agents.
- ★ Relies on finite set of a priori known policies
- ★ Galceran, E., Cunningham, A. G., Eustice, R. M., & Olson, E. (2017). **Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction: Theory and experiment.** *Autonomous Robots*, 1-16.



Structure

- ✦ Modeling a car
- ✦ Planning
- ✦ Control
- ✦ Interaction with other Drivers
- ✦ Case Studies
 - ⑩ MATSim: Macroscopic Traffic Simulator
 - ⑩ Autonovi



Autonomous Driving: MATSim

- ★ Large-scale agent-based transport simulations
 - ⑩ Extendable modules
 - ⑩ High level agent behaviors/activities
 - ⑩ Capable of running massive simulations
 - ⑩ Analysis tools
 - ⑩ Suitable for road network planning, traffic analysis and routing



Structure

- ✦ Modeling a car
- ✦ Planning
- ✦ Control
- ✦ Interaction with other Drivers
- ✦ Case Studies
 - ⑩ MATSim: Macroscopic Traffic Simulator
 - ⑩ **Autonovi**



References

- ✦ Katrakazas, C., Quddus, M., Chen, W. H., & Deka, L. (2015). Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60, 416-442.
- ✦ Pendleton, S. D., Andersen, H., Du, X., Shen, X., Meghjani, M., Eng, Y. H., ... & Ang, M. H. (2017). Perception, Planning, Control, and Coordination for Autonomous Vehicles. *Machines*, 5(1), 6.
- ✦ Paden, B., Čáp, M., Yong, S. Z., Yershov, D., & Frazzoli, E. (2016). A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1), 33-55.
- ✦ Urmson, C., Baker, C., Dolan, J., Rybski, P., Salesky, B., Whittaker, W., ... & Darms, M. (2009). Autonomous driving in traffic: Boss and the urban challenge. *AI magazine*, 30(2), 17.
- ✦ Bojarski, M., Yeres, P., Choromanska, A., Choromanski, K., Firner, B., Jackel, L., & Muller, U. (2017). Explaining How a Deep Neural Network Trained with End-to-End Learning Steers a Car. *arXiv preprint arXiv:1704.07911*.
- ✦ MIT moral machine: <http://moralmachine.mit.edu/>
- ✦ U.S. National Highway Transportation Safety Administration: <https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety>



AutonoVi-Sim:

Modular Autonomous Vehicle Simulation Platform Supporting Diverse Vehicle Models, Sensor Configuration, and Traffic Conditions

Andrew Best, Sahil Narang, Lucas Pasqualin, Daniel Barber, Dinesh Manocha

University of North Carolina at Chapel Hill

UCF Institute for Simulation and Training

<http://gamma.cs.unc.edu/AutonoVi/>



Motivation

- 1.2 billion vehicles on the roads today
 - 84 million new vehicles in 2015
 - China: 24 m U.S.: 2.7m
 - India: 3.7 m S.E Asia: 3.8m
- Many markets expected to grow exponentially through 2030



New Delhi



Bangkok

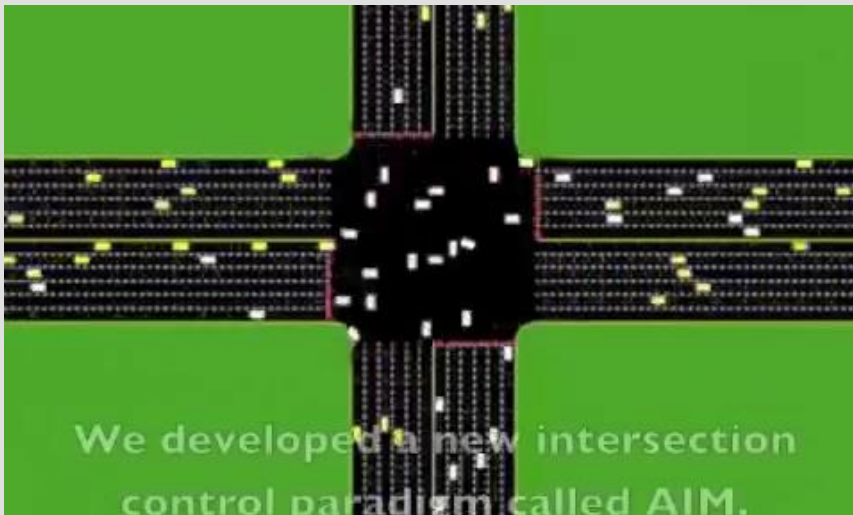
Motivation

- Majority of new vehicles in developing markets (30+ million)
- Limited infrastructure, loose traffic conventions
- Average vehicle life: 10+ years (17 years in U.S)



Motivation

- Long before autonomy will reach this:



Au et al. 2012



Kabbaj, TED 2016

Motivation

- It will deal with this:



Challenges

- Safety guarantees are critical
- Drivers, pedestrians, cyclists difficult to predict
- Road and environment conditions are dynamic
- Laws and norms differ by culture
- Huge number of scenarios

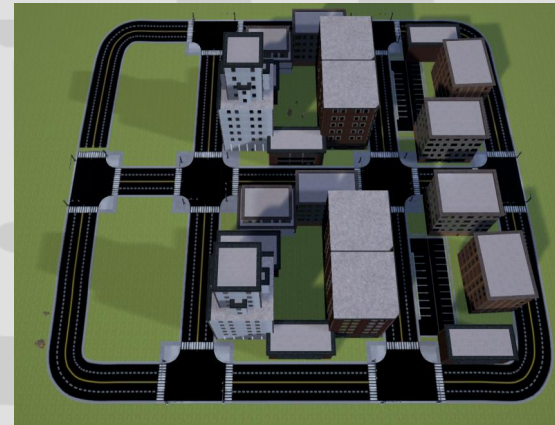


Challenges

- Development and testing of autonomous driving algorithms
 - On-road experiments may be hazardous
 - Closed-course experiments may limit transfer
 - High costs in terms of time and money
- Solution: develop and test robust algorithms in simulation
 - Test novel driving strategies & sensor configurations
 - Reduces costs
 - Allows testing dangerous scenarios
 - Vary traffic and weather conditions



Parking lot mock-up



Simulated city

Contributions

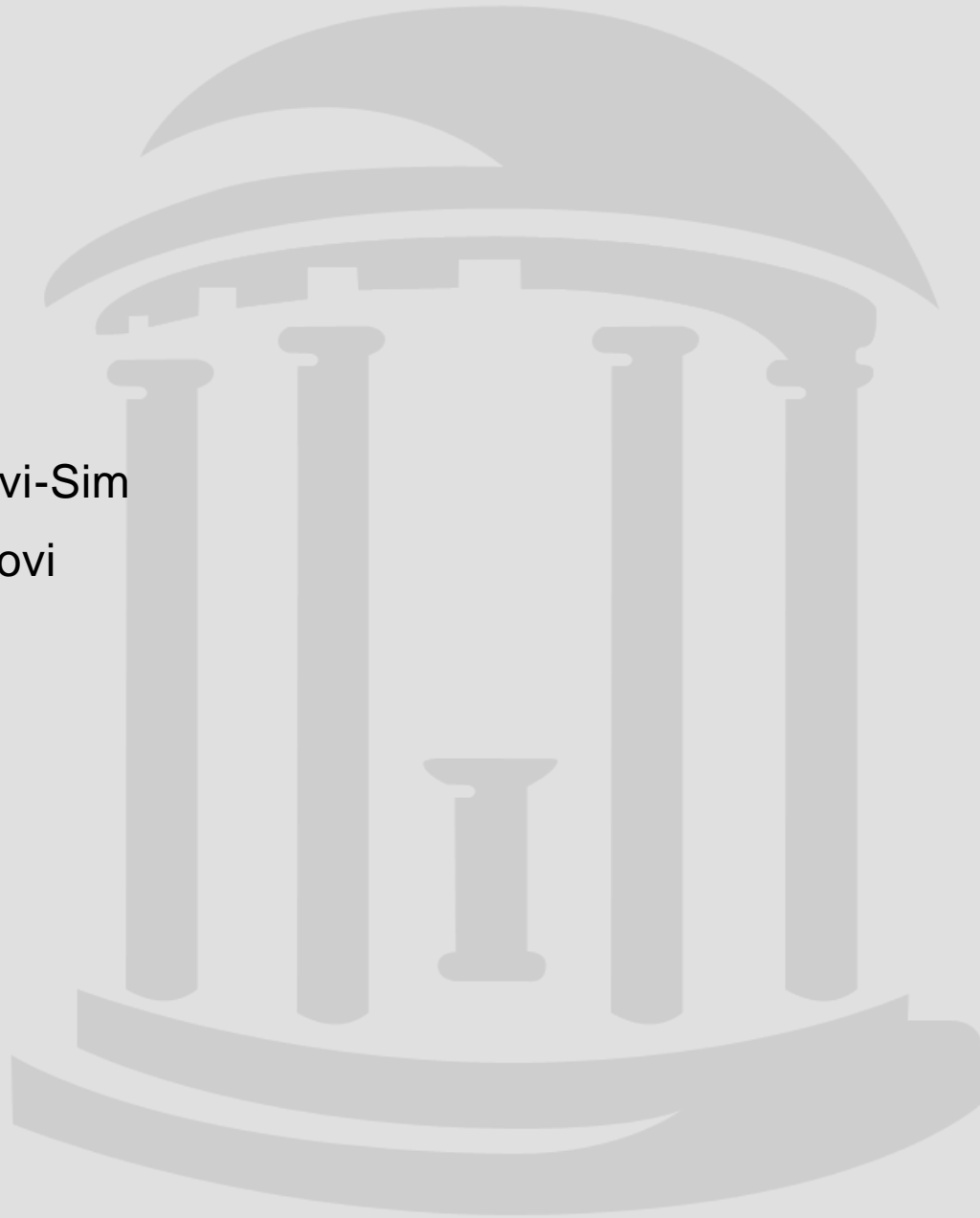
- **AutonoVi-Sim** : high fidelity simulation platform for testing autonomous driving algorithms
 - Varying vehicle types, traffic condition
 - Rapid Scenario Construction
 - Simulates cyclists and pedestrians
 - Modular Sensor configuration, fusion
 - Facilitates testing novel driving strategies

Contributions

- **AutonoVi**: novel algorithm for autonomous vehicle navigation
 - Collision-free, dynamically feasible maneuvers
 - Navigate amongst pedestrians, cyclists, other vehicles
 - Perform dynamic lane-changes for avoidance and overtaking
 - Generalizes to different vehicles through data-driven dynamics approach
 - Adhere to traffic laws and norms

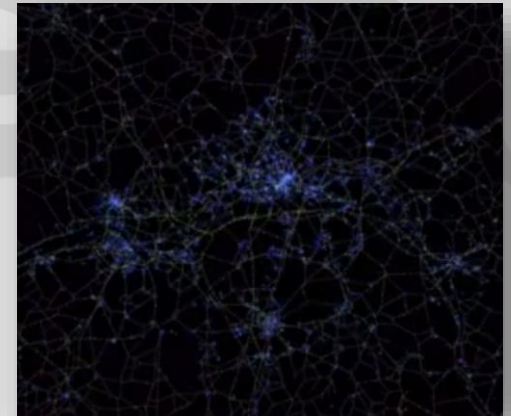
Overview

- Motivation
- **Related Work**
- Contributions:
 - Simulation Platform: Autonovi-Sim
 - Navigation Algorithm: Autonovi
- Results



Related work:

- Traffic Simulation
 - MATSim [Horni 2016], SUMO [krajzewicz 2002]
- Autonomous Vehicle Simulation
 - OpenAI Universe, Udacity
 - Waymo Carcraft, Righthook.io
- Simulation integral to development of many controllers & recent approaches [Katrakazas2015].



MATSim



SUMO

Related work:

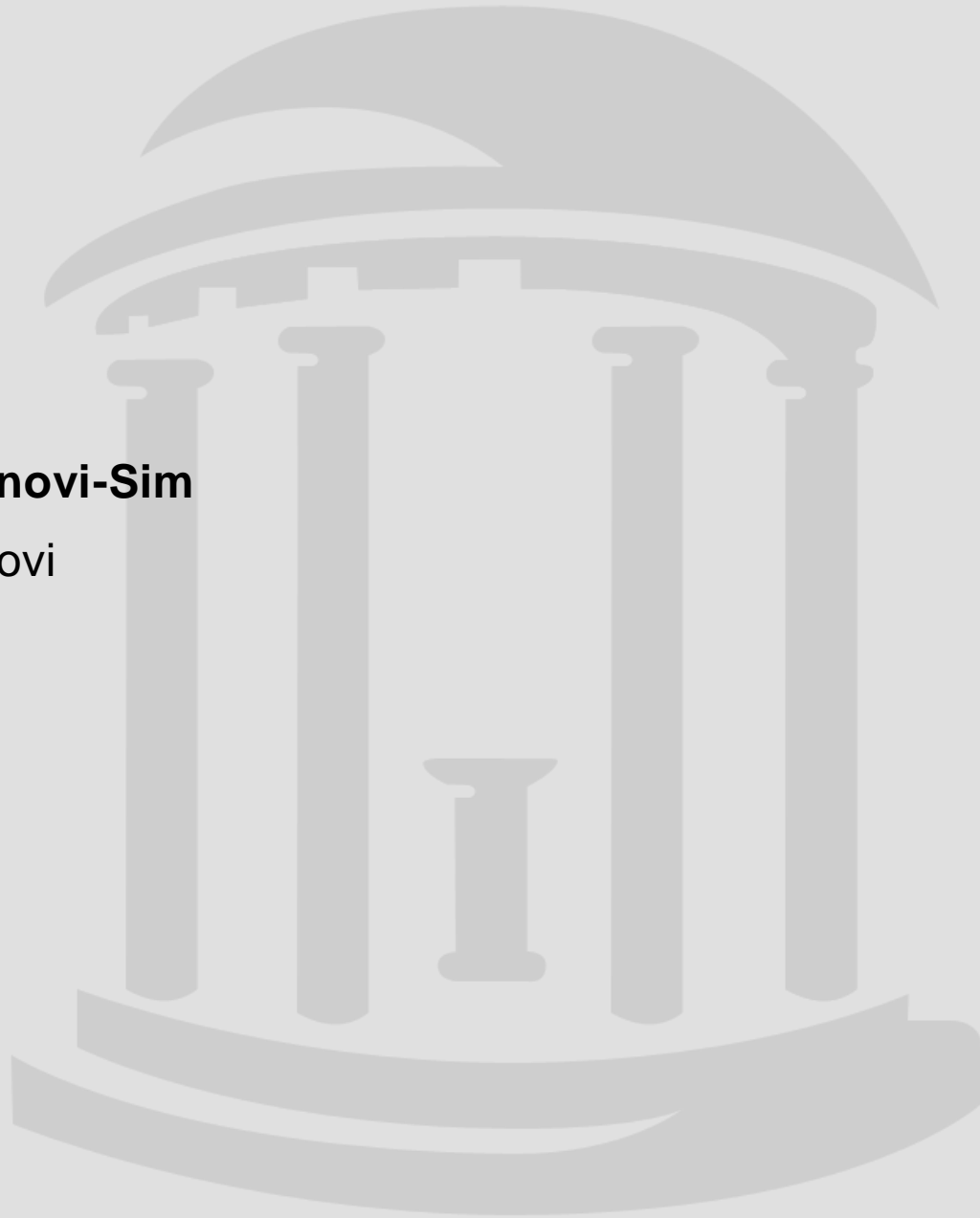
- Collision-free navigation
 - Occupancy grids [Kolski 2006], driving corridors [Hardy 2013]
 - Velocity Obstacles [Berg 2011], Control obstacles [Bareiss 2015], polygonal decomposition [Ziegler 2014], random exploration [Katrakazas 2015]
 - Lateral control approaches [Fritz 2004, Sadigh 2016]
- Generating traffic behaviors
 - Human driver model [Treiber 2006], data-driven [Hidas 2005], correct by construction [Tumova 2013], Bayesian prediction [Galceran 2015]

Related work:

- Modelling Kinematics and Dynamics
 - kinematic models [Reeds 1990, LaValle 2006, Margolis 1991]
 - Dynamics models [Borrelli 2005]
- Simulation for Vision Training
 - Grand Theft Auto 5 [Richter 2016, Johnson-Roberson 2017]

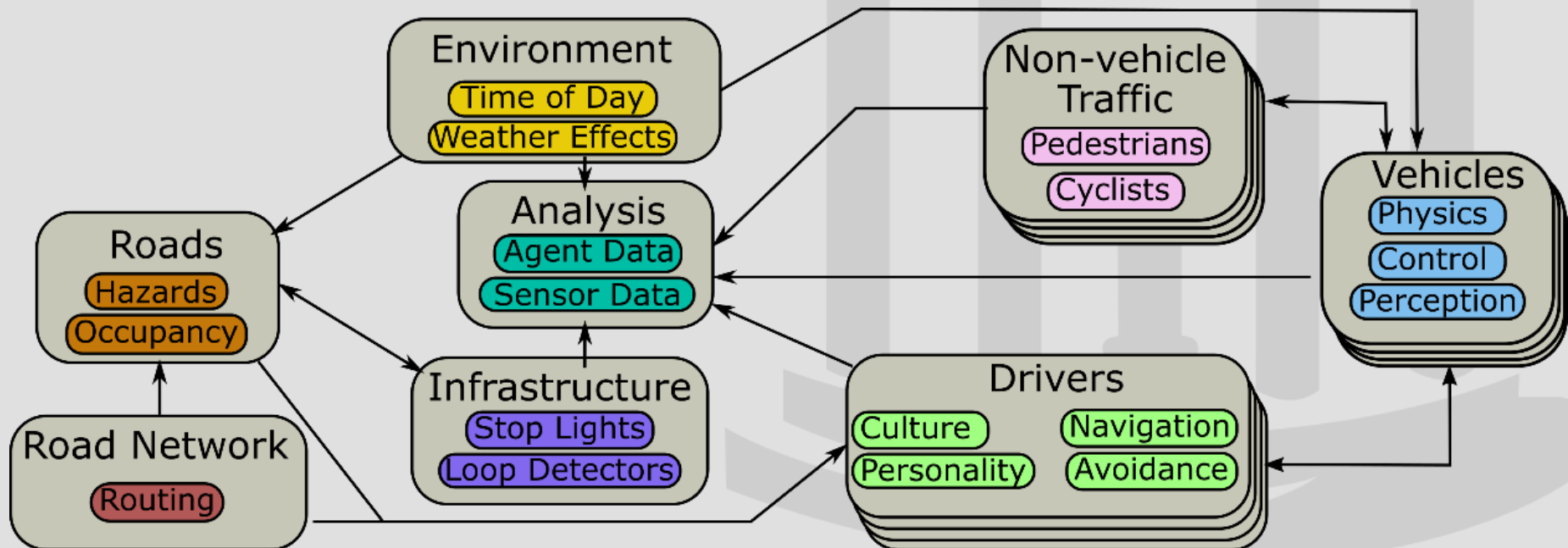
Overview

- Motivation
- Related Work
- **Contributions:**
 - **Simulation Platform: Autonovi-Sim**
 - Navigation Algorithm: Autonovi
- Results



Autonovi-Sim

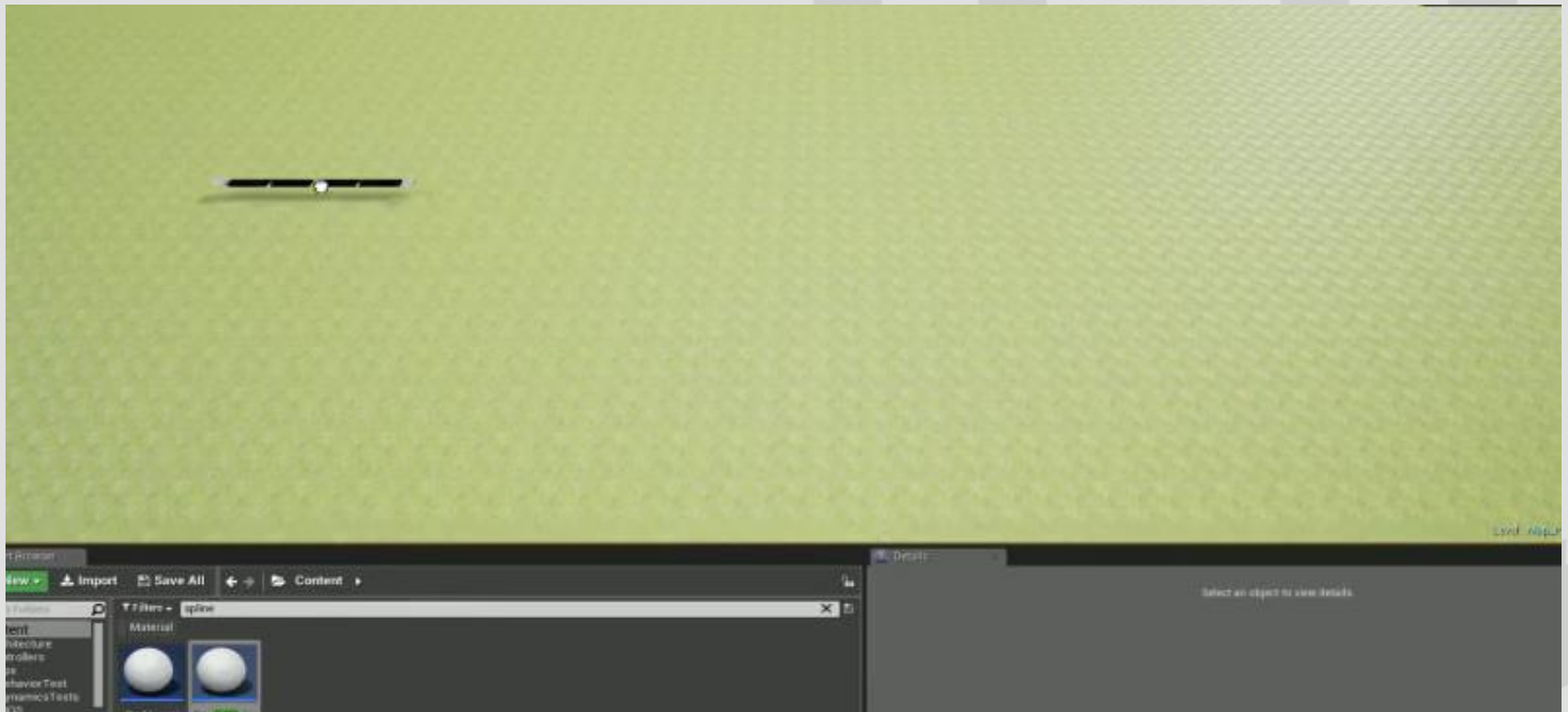
- Modular simulation framework for generating dynamic traffic conditions, weather, driver profiles, and road networks
- Facilitates novel driving strategy development



Autonovi-Sim: Roads & Road Network

- Roads constructed by click and drag
- Road network constructed automatically

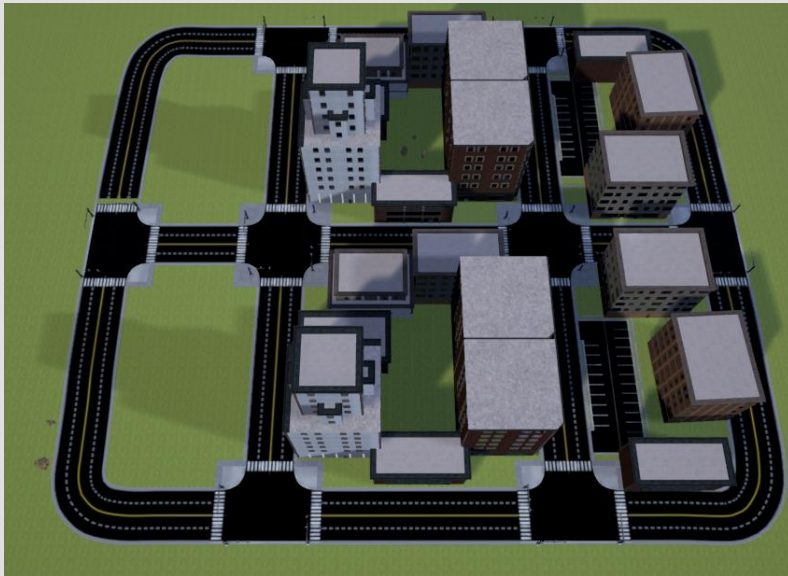
Roads
Hazards
Occupancy



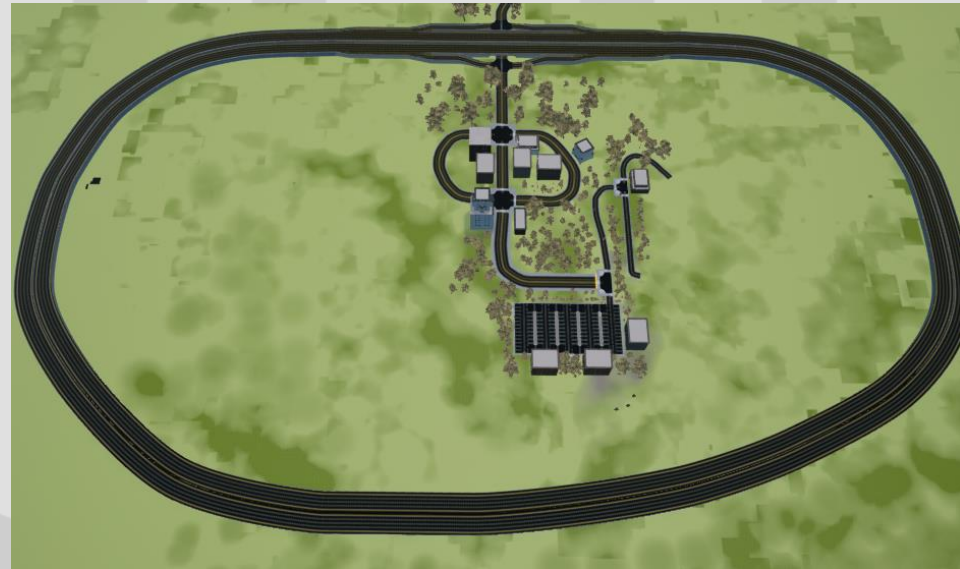
Road layouts

Autonovi-Sim: Roads & Road Network

- Construct large road networks with minimal effort
- Provides routing and traffic information to vehicles
- Allows dynamic lane closures, sign obstructions



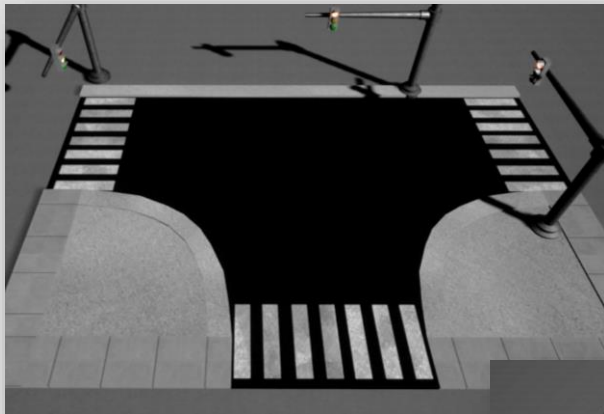
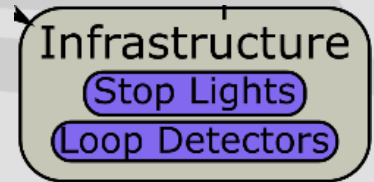
Urban Environment for pedestrian
& cyclist testing



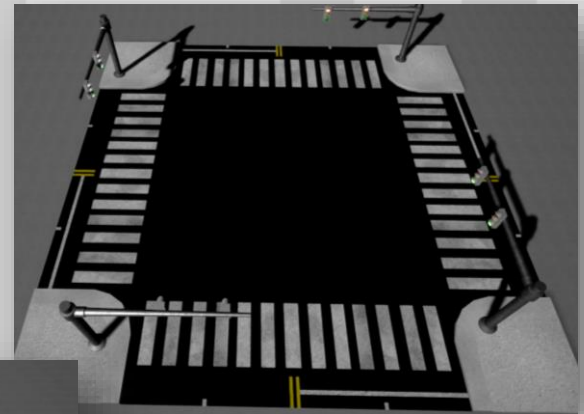
4 kilometer highway on and off loop

Autonovi-Sim: Infrastructure

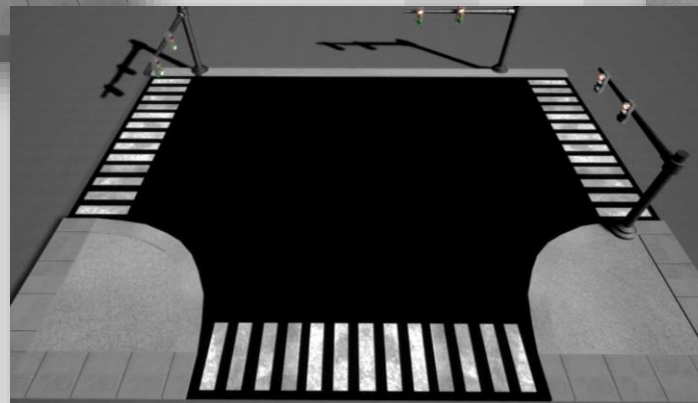
- Infrastructure placed as roads or overlays
- Provide cycle information to vehicles, can be queried and centrally controlled



3 way, one lane



4 way, two lane



3 way, two lane

Autonovi-Sim: Environment

- Goal: Testing driving strategies & sensor configuration in adverse conditions
- Simulate changing environmental conditions
 - Rain, fog, time of day
 - Modelling associated physical changes

Environment

Time of Day

Weather Effects



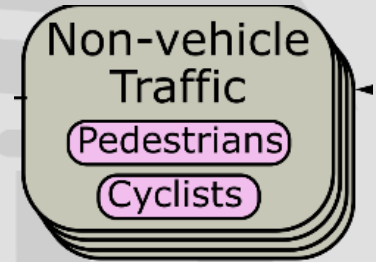
Fog reduces visibility



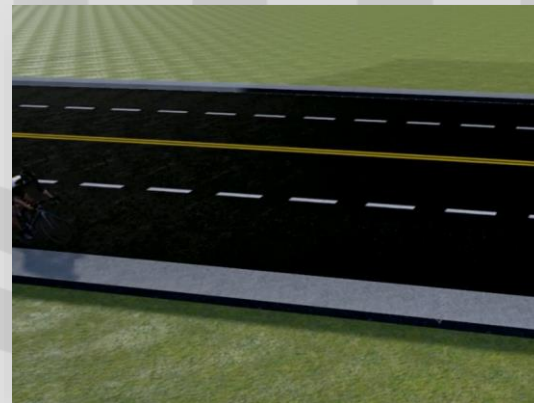
Heavy rain reduces traction

Autonovi-Sim: Non-vehicle Traffic

- Cyclists
 - operate on road network
 - Travel as vehicles, custom destinations and routing
- Pedestrians
 - Operate on roads or sidewalks
 - Programmable to follow or ignore traffic rules
 - Integrate prediction and personality parameters



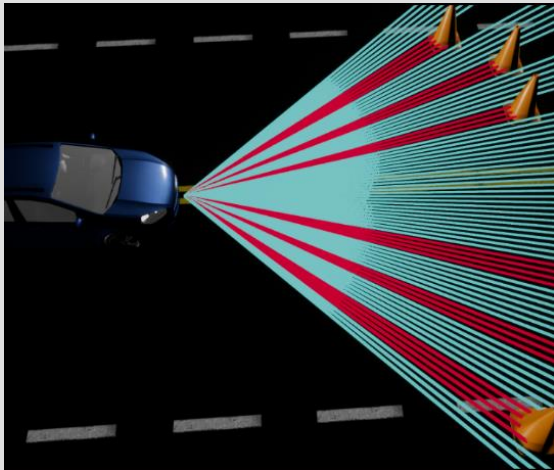
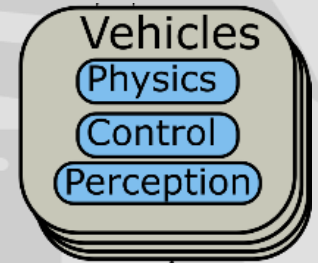
Pedestrian Motion



Cyclist Motion

Autonovi-Sim: Vehicles

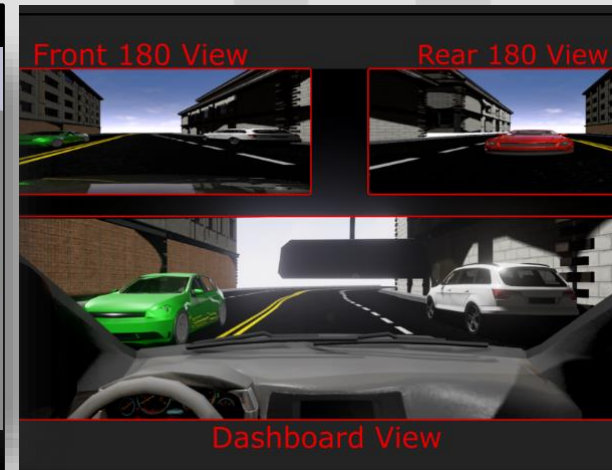
- Various vehicle profiles:
 - Size, shape, color
 - Speed / engine profile
 - Turning / braking
- Manage sensor information



Laser Range-finder



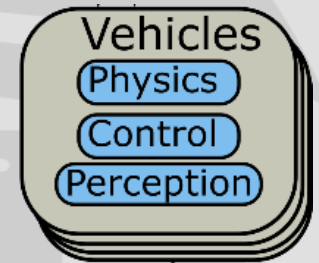
Multiple Vehicle Configurations



Multi-camera detector

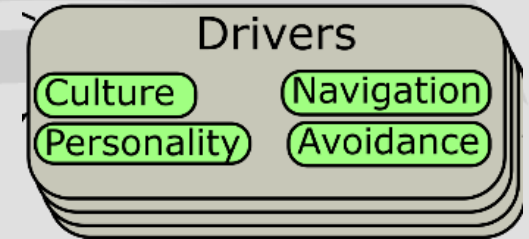
Autonovi-Sim: Vehicles

- Sensors placed interactively on vehicle
 - Configurable perception and detection algorithms



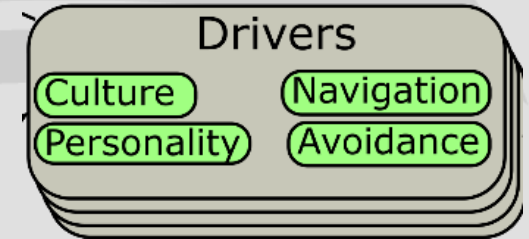
Autonovi-Sim: Drivers

- Control driving decisions
 - Fuse sensor information
 - Determine new controls (steering, throttle)
- Configurable parameters representing personality
 - Following distance, attention time, speeding, etc.
- Configure proportions of driver types
 - i.e. 50% aggressive, 50% cautious



Autonovi-Sim: Drivers

- 3 Drivers in AutonoVi-Sim
 - Manual
 - Basic Follower
 - AutonoVi



Manual Drive



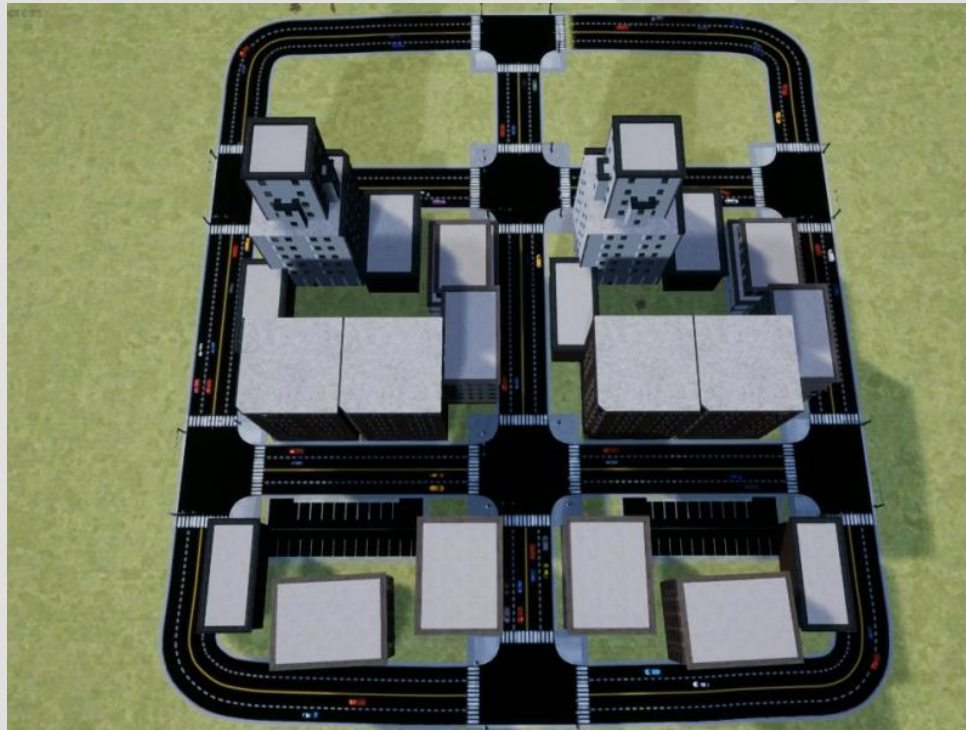
Basic Follower



AutonoVi

Autonovi-Sim: Results

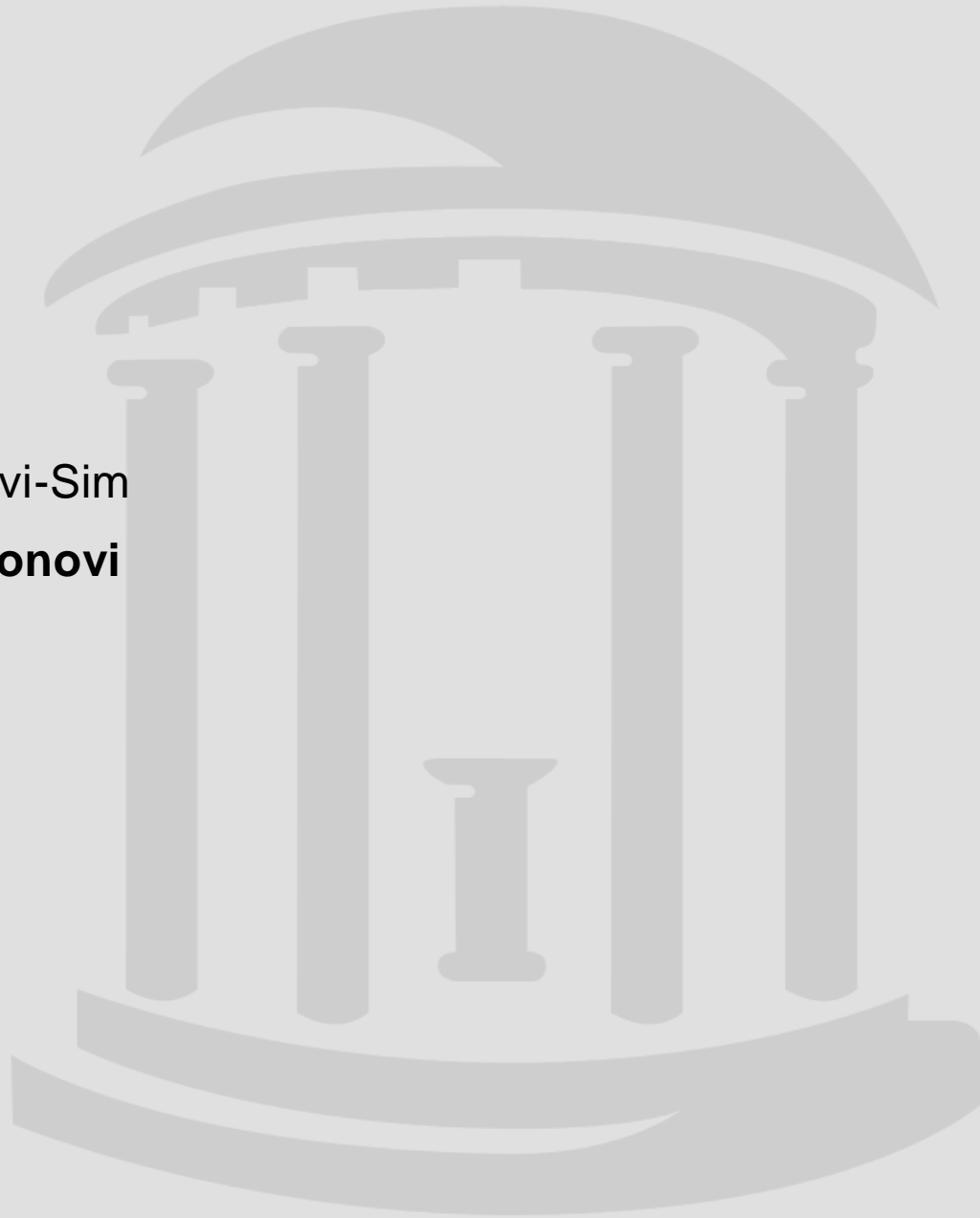
- Simulating large, dense road networks
- Generating data for analysis, vision classification, autonomous driving algorithms



50 vehicles navigating (3x)

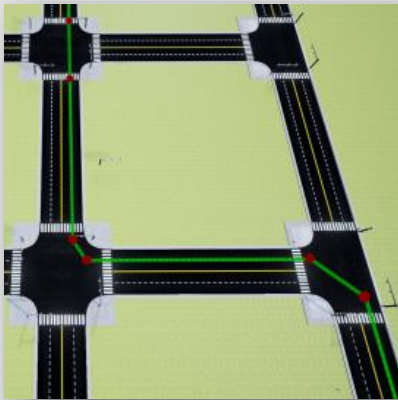
Overview

- Motivation
- Related Work
- **Contributions:**
 - Simulation Platform: Autonovi-Sim
 - **Navigation Algorithm: Autonovi**
- Results

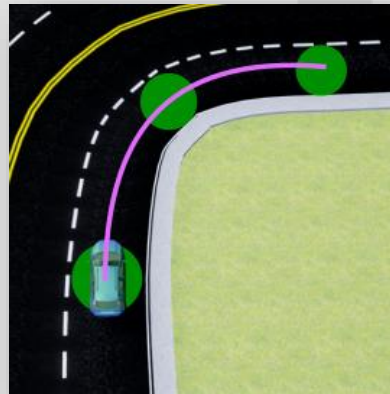


Autonovi

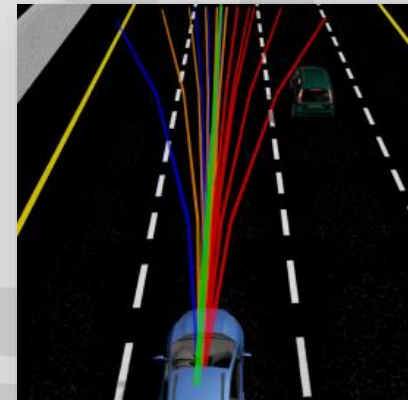
- Computes collision free, dynamically feasible maneuvers amongst pedestrians, cyclists, and vehicles
- 4 stage algorithm
 - Routing / GPS
 - Guiding Path Computation
 - Collision-avoidance / Dynamics Constraints
 - Optimization-based Maneuvering



GPS Routing



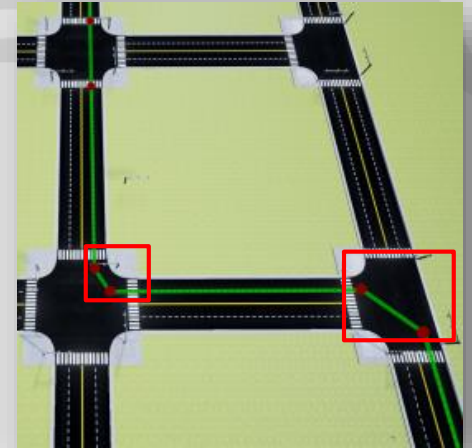
Guiding Path



Optimization-based
Maneuvering

Autonovi: Routing / GPS

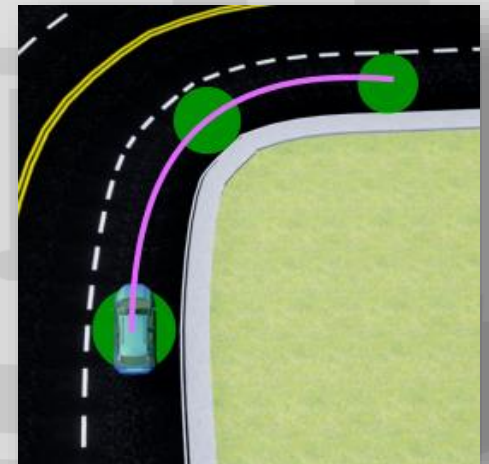
- Generates maneuvers between vehicle position and destination
- Nodes represent road transitions
- Allows vehicle to change lanes between maneuvers



GPS Routing

Autonovi: Guiding Path

- Computes “ideal” path vehicle should follow
- Respects traffic rules
- Path computed and represented as arc
- Generates target controls



Guiding Path

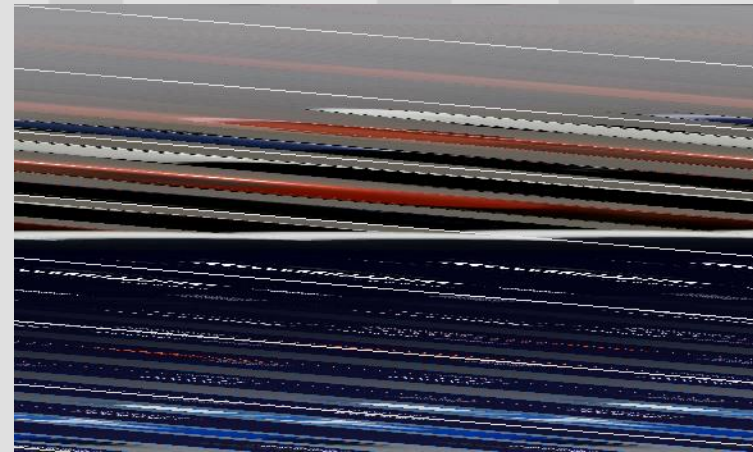
Autonovi: Collision Avoidance / Dynamics

- Control Obstacles [Bareiss 2015]
 - “Union of all controls that could lead to collisions with the neighbor within the time horizon, τ ”
 - Plan directly in control space (throttle, steering)
 - Construct “obstacles” for nearby entities
- Key principles / Assumptions
 - Reciprocity in avoidance (all agents take equal share)
 - Bounding discs around each entity
 - Controls / decisions of other entities are observable
 - New controls chosen as minimal deviation from target s. t. the following is not violated:

$$\forall (j \neq i, 0 \leq t < \tau) :: (\mathcal{O}_i \oplus \{\mathbf{q}_i(\mathbf{g}_i(t, \mathbf{x}_i, \mathbf{u}_i + \Delta \mathbf{u}_i))\}) \cap (\mathcal{O}_j \oplus \{\mathbf{q}_j(\mathbf{g}_j(t, \mathbf{x}_j, \mathbf{u}_j + \Delta \mathbf{u}_j))\}) = \emptyset$$

Autonovi: Collision Avoidance / Dynamics

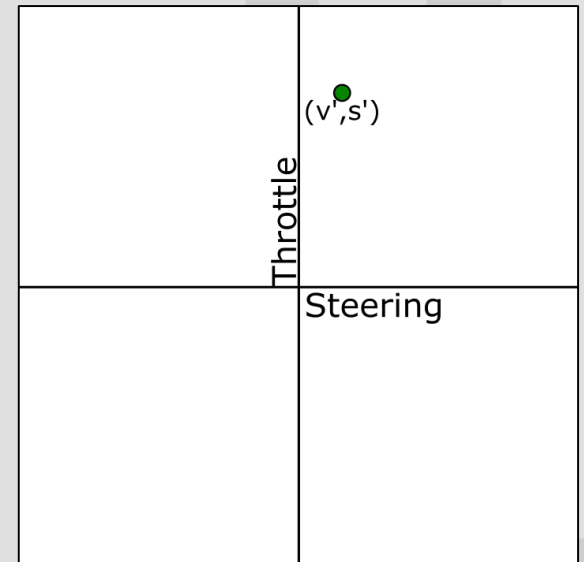
- Goal: Augment control obstacles with dynamics constraints
- Generate dynamics profile for vehicles through profiling
 - repeated simulation for each vehicle testing control inputs
- Represent underlying dynamics without specific model
- Gather data to generate approximation functions for non-linear vehicle dynamics
 - $S(\mu)$: target controls are safe given current vehicle state
 - $A(\mu)$: Expected acceleration given effort and current state
 - $\Phi(\mu)$: Expected steering change given effort and current state



Dynamics Profile Generation

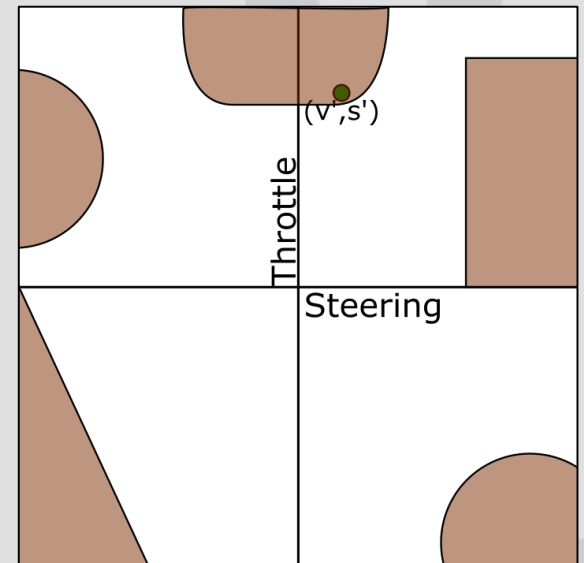
Autonovi: Collision Avoidance / Dynamics

- Augmented Control Obstacles
 - Reciprocity is not assumed from others
 - Use tightly fitting bounding polygons
 - Do not assume controls of others are observable
 - New controls chosen from optimization stage



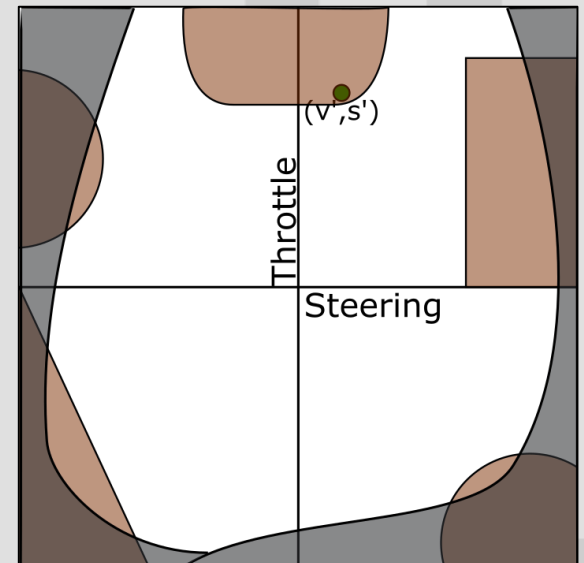
Autonovi: Collision Avoidance / Dynamics

- Augmented Control Obstacles
 - Reciprocity is not assumed from others
 - Use tightly fitting bounding polygons
 - Do not assume controls of others are observable
 - New controls chosen from optimization stage
- Obstacles constructed from avoidance



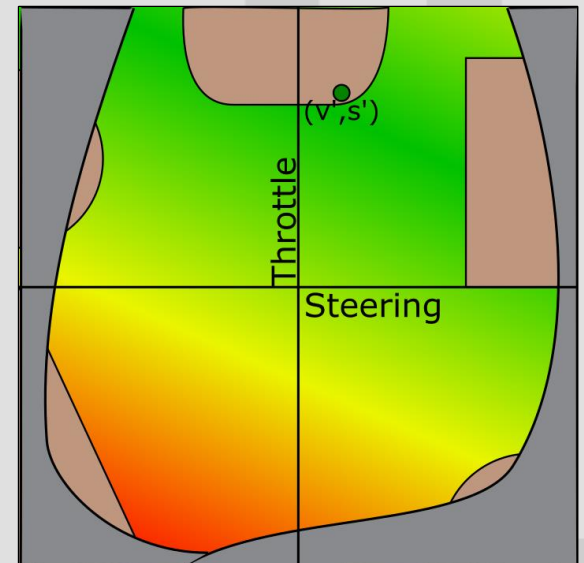
Autonovi: Collision Avoidance / Dynamics

- Augmented Control Obstacles
 - Reciprocity is not assumed from others
 - Use tightly fitting bounding polygons
 - Do not assume controls of others are observable
 - New controls chosen from optimization stage
- Obstacles constructed from avoidance
- Obstacles constructed from dynamics



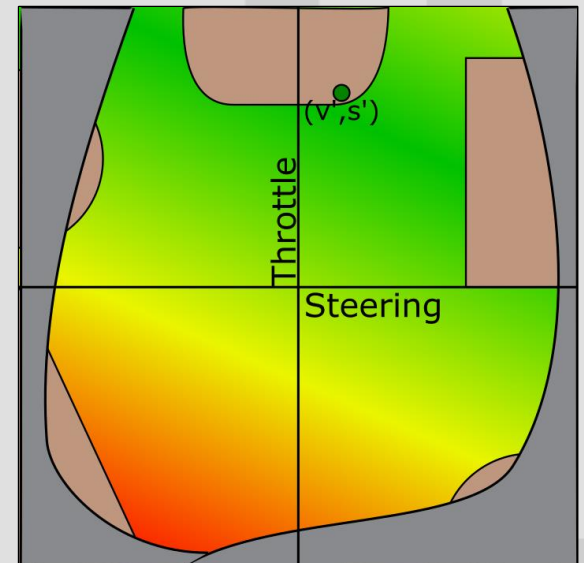
Autonovi: Collision Avoidance / Dynamics

- Augmented Control Obstacles
 - Reciprocity is not assumed from others
 - Use tightly fitting bounding polygons
 - Do not assume controls of others are observable
 - New controls chosen from optimization stage
- Obstacles constructed from avoidance
- Obstacles constructed from dynamics
- New velocity chosen by cost-optimization



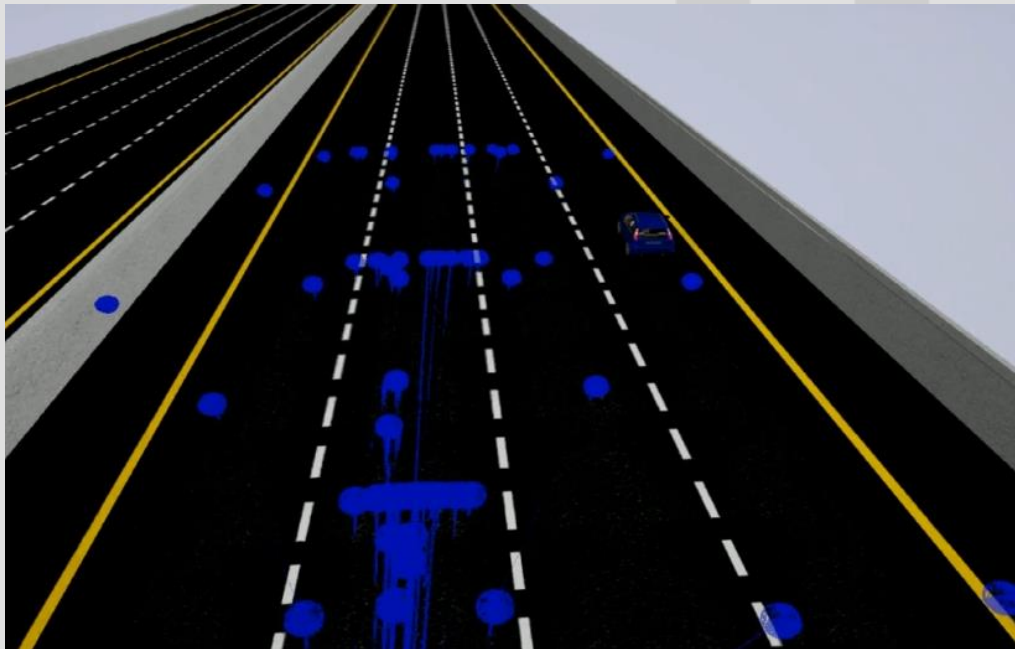
Autonovi: Collision Avoidance / Dynamics

- Advantages of augmented control obstacles:
 - Free-space is guaranteed feasible and safe
 - Conservative linear constraints from surface of obstacles
- Disadvantages:
 - Closed-form of surface may not exist
 - Space may be non-convex
 - Computationally expensive



Autonovi: Collision Avoidance / Dynamics

- Sampling approach
 - Construct candidate controls via sampling near target controls
- Evaluate collision-avoidance and dynamics constraints
 - Forward integrate safe controls to generate candidate trajectories
- Choose “optimal” control set in optimization stage



Autonovi: Optimization-Based Maneuvering

- Choose “optimal” controls through multi-objective cost function
- Path (velocity, drift, progress)
- Comfort (acceleration, yaw)
- Maneuver (lane change, node distance)
- Proximity (cyclists, vehicle, pedestrians)

$$C = \sum_{i=0}^I c_{path}(i) + c_{cmft}(i) + c_{mnvr}(i) + c_{prox}(i)$$

Autonovi: Optimization-Based Maneuvering

- Choose “optimal” controls through multi-objective cost function
- Path (velocity, drift, progress)
- Comfort (acceleration, yaw)
- **Maneuver (lane change, node distance)**
 - Static cost for lane changes
 - Cost inverse to distance if vehicle occupies incorrect lane as maneuver approaches
- Proximity (cyclists, vehicle, pedestrians)

$$C = \sum_{i=0}^I c_{path}(i) + c_{cmft}(i) + c_{mnvr}(i) + c_{prox}(i)$$

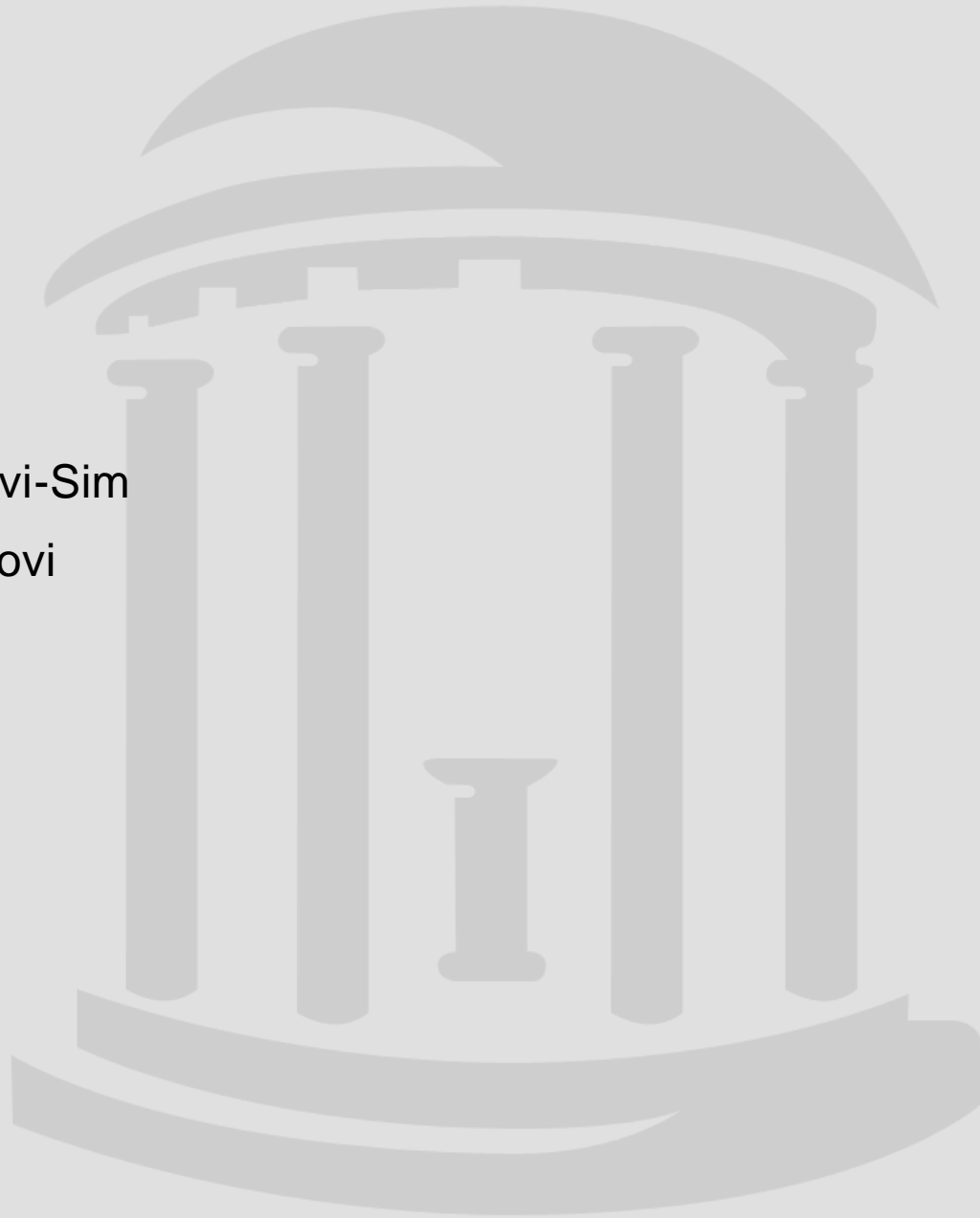
Autonovi: Optimization-Based Maneuvering

- Choose “optimal” controls through multi-objective cost function
- Path (velocity, drift, progress)
- Comfort (acceleration, yaw)
- Maneuver (lane change, node distance)
- **Proximity (cyclists, vehicle, pedestrians)**
 - Configurable cost per entity type
 - Generates safe passing buffers

$$C = \sum_{i=0}^I c_{path}(i) + c_{cmft}(i) + c_{mnvr}(i) + c_{prox}(i)$$

Overview

- Motivation
- Related Work
- Contributions:
 - Simulation Platform: Autonovi-Sim
 - Navigation Algorithm: Autonovi
- **Results**



Results: Sudden Hazards @ 20 mph

- Vehicle responds quickly to sudden hazards
 - Braking and swerving to avoid collisions



Results: Sudden Hazards @ 60 mph

- Vehicle responds quickly to sudden hazards
 - Respects unique dynamics of each car



Results: Jaywalking Pedestrian

- Vehicle accounts for pedestrians and comes to a stop



Results: Jaywalking Pedestrian

- Vehicle accounts for pedestrians and comes to a stop
 - Respects unique dynamics of each car



Results: Passing Cyclists

- Vehicle changes lanes to safely pass cyclist



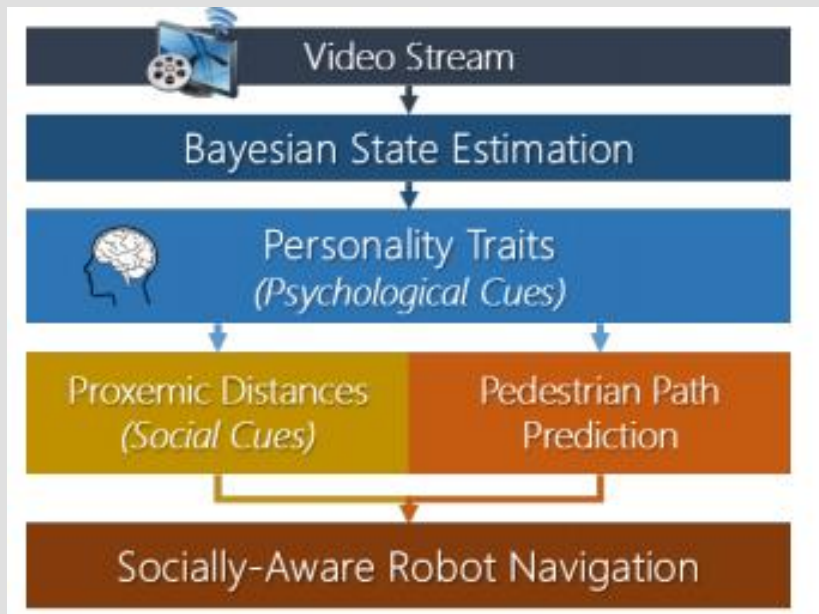
Results: Passing Cyclists

- Vehicle changes lanes to safely pass cyclist
 - Lane change only when possible

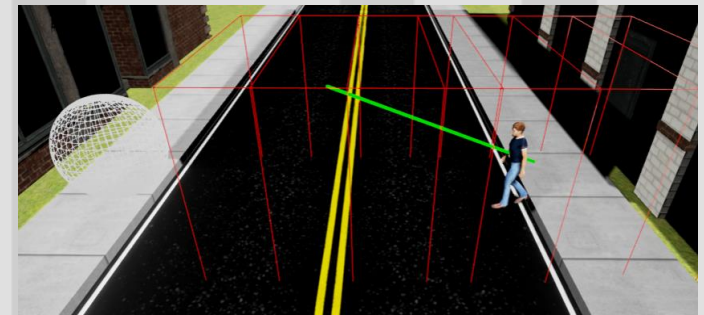


Results: Next Steps

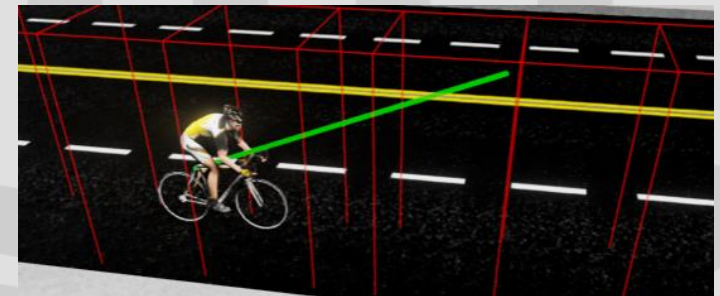
- Integrating prediction compensates for uncertainty
- Leverage behavior models to predict behavior
 - Bayesian Behavior prediction
 - Personality Trait Theory



SocioSense approach to pedestrian response



Predicting Future Pedestrian Trajectories



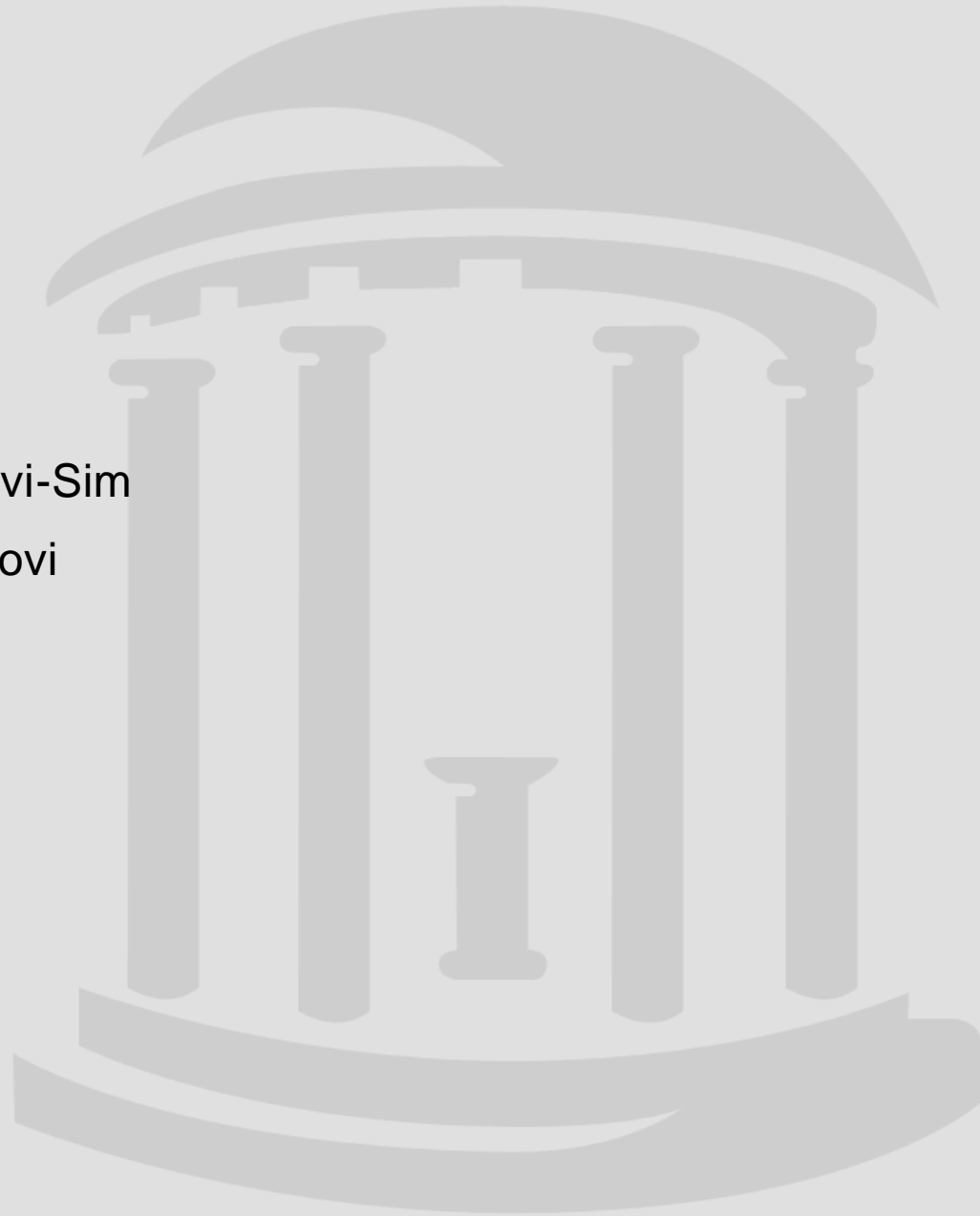
Predicting Future Cyclist Trajectories

Results: Next Steps

- Using real-world training data, behaviors can be optimized to improve realism
 - Ex: Drivers behave more like human drivers
 - Ex: Infrastructure tuned to specific real patterns
- Vehicle sensors can be similarly calibrated

Overview

- Motivation
- Related Work
- Contributions:
 - Simulation Platform: Autonovi-Sim
 - Navigation Algorithm: Autonovi
- Results
- **Conclusion**



Conclusions

- Simulation of hundreds of vehicles, pedestrians, cyclists
- Configurable sensors and driver behavior
- Collision-free, dynamically feasible maneuvers
- Perform dynamic lane-changes for avoidance and overtaking
- Generalizes to different vehicles through data-driven dynamics profiling

Limitations

- Data-driven dynamics rely on simulation
- Reliance on perfect sensing
- Parameter weights manually optimized
- Manual Sensor Calibration

Future Work

- Data-driven parameter weight learning
- Validation of dynamics modelling
- Improve generation and handling of sensor uncertainty
- Behavior prediction for nearby entities
- Combine with road-network data to generate scenarios
- Additional sensor implementations

References

- [1] Julius Ziegler, Philipp Bender, Thao Dang, and Christoph Stiller. Trajectory planning for Bertha - A local, continuous method. *The International Journal of Robotics Research*, 35(April):450–457, 2014.
- [2] Pendleton et al. Perception, Planning, Control, and Coordination for Autonomous Vehicles. *Machines*, 5(1):6, 2017.
- [3] Ziegler et al. Making bertha drive-an autonomous journey on a historic route. *IEEE Intelligent Transportation Systems Magazine*, 6(2):8–20, 2014.
- [4] F. Borrelli, P. Falcone, T. Keviczky, J. Asgari, and D. Hrovat. MPCbased approach to active steering for autonomous vehicle systems. *International Journal of Vehicle Autonomous Systems*, 3(2/3/4):265, 2005.
- [5] Andreas Eidehall, Jochen Pohl, Fredrik Gustafsson, and Jonas Ekmark. Toward autonomous collision avoidance by steering. *IEEE Transactions on Intelligent Transportation Systems*, 8(1):84–94, 2007.
- [6] Martin Distner, Mattias Bengtsson, Thomas Broberg, and Lotta Jakobsson. City Safety- A System Addressing Rear-End Collisions At Low Speeds. *21st Enhanced Safety Vehicles Conference*, pages 1–7, 2009.
- [7] Turri et al. Linear model predictive control for lane keeping and obstacle avoidance on low curvature roads. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, (Itsc)*:378–383, 2013.
- [8] Christos Katrakazas, Mohammed Quddus, Wen-Hua Chen, and Lipika Deka. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60:416–442, 2015.
- [9] Mohammad Saifuzzaman and Zuduo Zheng. Incorporating humanfactors in car-following models: A review of recent developments and research needs. *Transportation Research Part C: Emerging Technologies*, 48:379–403, 2014.
- [10] Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, New York, NY, USA, 2006.
- [11] James Reeds and Lawrence Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 145(2):367–393, 1990. [12] Donald L. Margolis and Jahan Asgari. Multipurpose models of vehicle dynamics for controller design. In *SAE Technical Paper*. SAE International, 09 1991.
- [13] Sascha Kolski, D. Ferguson, Mario Bellino, and Roland Siegwart. Autonomous Driving in Structured and Unstructured Environments. In *2006 IEEE Intelligent Vehicles Symposium*, pages 558–563. IEEE, 2006.
- [14] Kuwata et al. Real-time motion planning with applications to autonomous urban driving. *IEEE Transactions on Control Systems Technology*, 17(5):1105–1118, 2009.
- [15] Jason Hardy and Mark Campbell. Contingency Planning Over Probabilistic Obstacle Predictions for Autonomous Road Vehicles. *IEEE Transactions on Robotics*, 29(4):913–929, aug 2013.
- [16] Enric Galceran, Ryan M Eustice, and Edwin Olson. Toward Integrated Motion Planning and Control using Potential Fields and Torque-based Steering Actuation for Autonomous Driving. *IEEE Intelligent Vehicles Symposium, (lv)*, 2015.

References

- [17] Dorsa Sadigh, Shankar Sastry, Sanjit A Seshia, and Anca D Dragan. Planning for Autonomous Cars that Leverage Effects on Human Actions. *Proceedings of Robotics: Science and Systems*, 2016.
- [18] Enric Galceran, Alexander G Cunningham, Ryan M Eustice, and Edwin Olson. Multipolicy Decision-Making for Autonomous Driving via Changepoint-based Behavior Prediction. *Robotics: Science and Systems*, 2015.
- [19] Julius Ziegler, Moritz Werling, and Joachim Schroder. Navigating car- like robots in unstructured environments using an obstacle sensitive cost function. *IEEE Intelligent Vehicles Symposium, Proceedings*, pages 787–791, 2008.
- [20] Gabriel M. Hoffmann, Claire J. Tomlin, Michael Montemerlo, and Sebastian Thrun. Autonomous automobile trajectory tracking for offroad driving: Controller design, experimental validation and racing. *Proceedings of the American Control Conference*, pages 2296–2301, 2007.
- [21] H. Fritz, A. Gern, H. Schiemenz, and C. Bonnet. CHAUFFEUR Assistant: a driver assistance system for commercial vehicles based on fusion of advanced ACC and lane keeping. *IEEE Intelligent Vehicles Symposium*, 2004, (Vc):495–500, 2004.
- [22] Jur Van Den Berg, Jamie Snape, Stephen J. Guy, and Dinesh Manocha. Reciprocal collision avoidance with acceleration-velocity obstacles. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3475–3482, 2011.
- [23] Daman Bareiss and Jur van den Berg. Generalized reciprocal collision avoidance. *The International Journal of Robotics Research*, 34(12):1501–1514, oct 2015.
- [24] Hao Sun, Weiwen Deng, Sumin Zhang, Shanshan Wang, and Yutan Zhang. Trajectory planning for vehicle autonomous driving with uncertainties. *ICCSS 2014 - Proceedings: 2014 International Conference on Informative and Cybernetics for Computational Social Systems*, pages 34–38, 2014.
- [25] Martin Treiber, Arne Kesting, and Dirk Helbing. Delays, inaccuracies and anticipation in microscopic traffic models. *Physica A: Statistical Mechanics and its Applications*, 360(1):71–88, 2006.
- [26] Peter Hidas. Modelling vehicle interactions in microscopic simulation of merging and weaving. *Transportation Research Part C: Emerging Technologies*, 13(1):37–62, 2005. [27] Jana Tumova, Gavin C Hall, Sertac Karaman, Emilio Frazzoli, and Daniela Rus. Least-violating control strategy synthesis with safety rules. In *Proceedings of the 16th international conference on Hybrid systems: computation and control*, pages 1–10. ACM, 2013.
- [28] Bo Chen and Harry H. Cheng. A review of the applications of agent technology in traffic and transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 11(2):485–497, 2010.
- [29] Geiger et al. Team AnnieWAY's Entry to the 2011 Grand Cooperative Driving Challenge. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1008–1017, 2012.
- [30] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. springer, 2009.
- [31] Jean paul Laumond, S. Sekhavat, and F. Lamiraux. Guidelines in nonholonomic motion planning for mobile robots. In *ROBOT MOTION PLANNING AND CONTROL*, pages 1–53. Springer-Verlag, 1998. [32] Fredrik Gustafsson. Slip-based tire-road friction estimation. *Automatica*, 33(6):1087–1099, 1997.
- [34] Aniket Bera, Tanmay Randhavane, and Dinesh Manocha. Sociosense: Robot navigation amongst pedestrians with social and psychological constraints. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017.