Another course: **Autonomous Mobile Robotics**
http://wavelab.uwaterloo.ca/sharedata/ME597/

Driving tasks:
- Perceiving the environment
- Planning how to reach from point A to B
- Controlling the car

Operational Design Domain (ODD)
The purpose of ODD monitoring is to determine whether or not the ADS is in a situation that it was designed to handle safely. The ODD can also be described as the “functional system boundary”. For example, if the ODD contains only unsignalized intersections, the ADS must not enter intersections controlled by traffic lights. If ODD monitoring determines that the ADS has violated, or is about to violate, the bounds of the ODD, then it is expected to initiate a Dynamic Driving Task (DDT) fallback in order to achieve a minimal risk condition. A DDT fallback for a level 3 ADS is human intervention and a DDT fallback for a level 4 or 5 ADS is executed by the ADS itself.
How to classify driving system automation?
- Driver attention requirement
- Driver action requirement
- What exactly makes up a driving task?

What makes up a driving task?
- Lateral control: steering
- Longitudinal control: braking, acceleration
- Object and Event Detection and Response: detection and reaction
- Planning: long term and short term
- Misc: signal indicators, interacting with other drivers

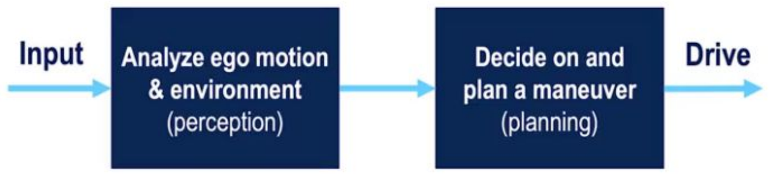
Autonomous capabilities: automate the driving task
- Automated lateral control
- Automated longitudinal control
- Complete vs Restricted ODD
- OEDR
    - Automatic emergency response
    - Driver supervision

Level 0: no automation: regular vehicles
Level 1: Driving assistance: either longitudinal control or lateral control, but not both
Examples: Adaptive Cruise Control: can control speed, but driver steers
Lane keeping:
Level 2: Partial driving automation:
Both longitudinal control and lateral control
Examples: GM super cruise and Nissan ProPilot Assist
Level 3: Conditional driving automation: driver can take their attention off compared to level 2
Includes OEDR
Example: Audi A8 Sedan

Level 4: High Driving Automation
Handles emergencies autonomously, driver can entirely focus on other tasks.
But limited ODD
Level 5: Fully Driving Automation
Unlimited ODD

Environment Perception

Make sense of the environment and ourselves.
Two major tasks: identification and understanding motion
Why? To inform our driving decisions


Input
Analyze ego motion & environment (perception)
Decide on and plan a maneuver (planning)
Drive

Goals for perception
- Identify static objects: Road and lane markings, curbs, traffic lights, road signs, construction signs, obstructions
- Identify dynamic objects: vehicles, pedestrians, cyclists
- Ego localization: Position, velocity, acceleration, orientation and angular motion

Challenges to perceptions
- Robust detection and segmentation
- Sensor uncertainty: GPS, lidar and radar
- Occlusion, reflection
- Illumination, lens flare
- Weather precipitation

Planning

Making decisions
- Long term planning: how to navigate from NYC to LA? Mission plan.
- Short term policy: can I change my lane to the lane right of me? Can I pass the intersection and join the left road?
- Intermediate decisions/reactions: involves control and trajectory planning
    - Can I stay on track on this curved road?
    - Accelerate or brake by how much?

Example:
Taking a left turn in an intersection
- It involves many short-term planning decisions.
- There are situations arise on the way: object and event detection and response, another vehicle stops in front and we need to make room for the other vehicle. If the stop line is not marked, we want to approximate and infer the implied where the stopline is. All the

decisions are immediate decisions and require safe reaction to the planning system.The end result is an exploding list of possible decisions to evaluate on different timescales, even for a simple left turn scenario. This amounts to talking about different cases for the same intersection crossing or scenarios. In each scenario, we need a consistent set of choices to be evaluated in real time and updated as new information about the scene becomes available. Furthermore, because decisions to change lanes affect where to drive and which cars to regulate our position relative to, even a seemingly simple driving scenario requires three or four levels of decisions, and must then still be executed with careful vehicle control.

- Simple maneuver, yet it takes 3-4 levels of decisions and control to execute
- It takes many rules to drive
    - Safely
    - Efficiently
    - Following all traffic rules
    - Only following rules that everyone else is following
- Driving decision making is complicated

Rule based planning

- Decision trees
- We have rules to take into account the current state of ego and other objects and give decisions.
- Examples:
    - If there is a pedestrian on the road, stop
    - If speed limit changes, adjust speed to match it.

Predictive planning

- Make predictions about other vehicles and how they are moving. Then use these predictions to inform our decisions.
- Examples:
    - A cars has stopped for 10 seconds, and might be going to stop for another 10 seconds
    - Pedestrian is jaywalking. She will enter our lane by the time we reach her.

Sensors

- Sensors: device that measures and detects a property of the environment or changes to a property
- Categories:
    - Exteroceptive: detect the environment/ surroundings
    - Proprioceptive: internal sensors/ one’s own

Exteroceptive sensors

Sensors for perception: **Camera**

- Essential for correctly perceiving environment
- Comparison metrics:
    - Resolution: resolution is the number of pixels that create the image.

    - Field of view: The field of view is defined by the horizontal and vertical angular extent that is visible to the camera, and can be varied through lens selection and zoom.
    - Dynamic range: The dynamic range of the camera is the difference between the darkest and the lightest tones in an image. High dynamic range is critical for self-driving vehicles due to the highly variable lighting conditions encountered while driving especially at night.
- Trade-off between resolution and FOV
- Focal length, depth of field and frame rate
- Stereo camera: enables depth estimation from image data, which produces a disparity of a scene

LIDAR which stands for light detection and ranging sensor. LIDAR sensing involves shooting light beams into the environment and measuring the reflected return. It details 3D scene geometry from LIDAR point cloud

- Comparison metrics:
    - Number of beams
    - Points per second
    - Rotation rate
    - Field of view
    - Upcoming: solid state LIDAR

RADAR: radio detection and ranging. Robust object detection and relative speed estimation. It works in poor visibility like fog and precipitation.

- Comparison metrics:
    - Detection range
    - FOV
    - Position and speed accuracy
- Configurations
    - WFOV but short range
    - NFOV but long range

Ultrasonic: Originally so named for sound navigation and ranging, which measure range using sound waves.

- short -range inexpensive all-weather distance measurement
- Ideal for low-cost parking solutions
- Unaffected by lighting precipitation
- Comparison metrics
    - Range
    - FOV
    - cost

Proprioceptive sensors

- **Global Navigation Satellite System** (GNSS) and **Inertial measurement units** (IMU)
- Direct measure of ego vehicle states

        - position , velocity (GNSS)
        - Varying accuracies: RTK, PPP, DGPS
        - Angular rotation rate (IMU)
        - Acceleration: (IMU)
        - Heading (IMU, GPS)
    - **Wheel odometry sensor**
        - Tracks wheel velocities and orientations
        - Uses these to calculate overall speed and orientation of car
            - Speed accuracy
            - Position drift

Computing Hardware

Self-driving brain

- Take in all sensor data
- Computes actions
- Data processing (Drive PX/AGX, Intel Mobileye EyeQ)
- Image processing, object detection, mapping
    - GPUs
    - FPGAs: field programmable gate array
    - ASICs: Application specific integrated chip
    - Synchronization hardware
        - To sync different modules and provide a common clock

Hardware configuration design

Sensor coverage requirements for different scenarios

- Highway driving
- Urban driving

Overall coverage, blind spots

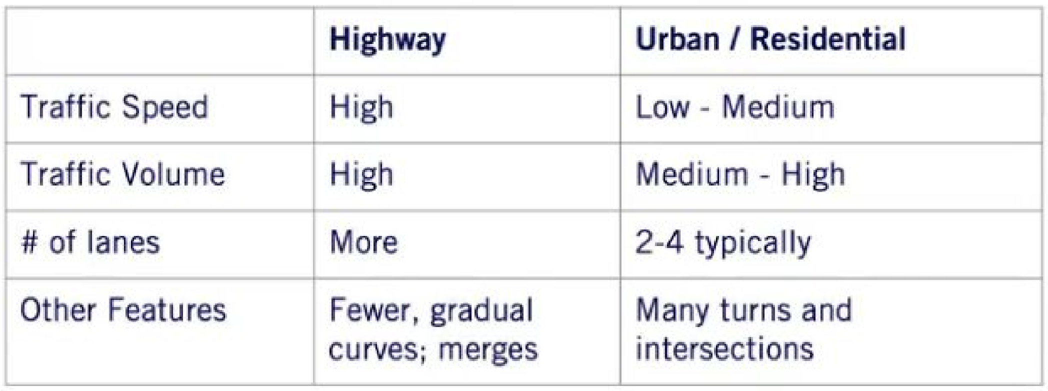
Sensor fusion: remember that all of these sensors come in different configurations and have different ranges in fields of view over which they can sense. They have some resolution that depends on the instrument specifics and field of view.

Assumptions:

- Aggressive deceleration: 5m/s^2
- Comfortable deceleration: 2m/s^2
- Stopping distance: d=v^2/2a

Where to place the sensors?

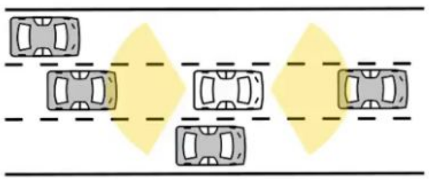
- Need sensors to support maneuvers within our ODD
- Two driving environments



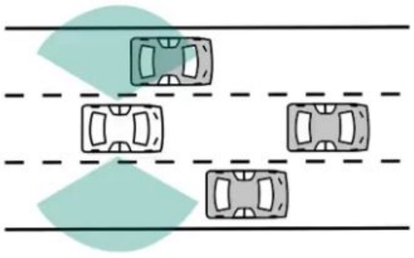
	Highway	Urban / Residential
Traffic Speed	High	Low - Medium
Traffic Volume	High	Medium - High
# of lanes	More	2-4 typically
Other Features	Fewer, gradual curves; merges	Many turns and intersections

3 major maneuvers in highway analysis

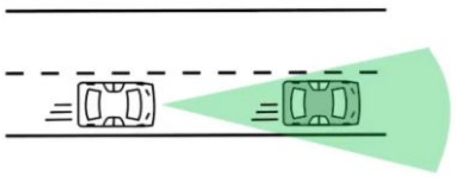
- Emergency stop
  - Longitude coverage: we are speeding at 120km/h and stopping distance should be 110 meters; aggressive deceleration



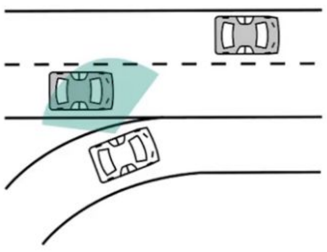
  - Lateral coverage: lane change to avoid a hard stop



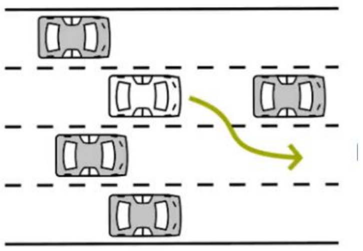
- Maintain speed: relative speeds are typically less than 30 kmph.
  - Longitude coverage: at least 100 meter in front. (16s for braking)



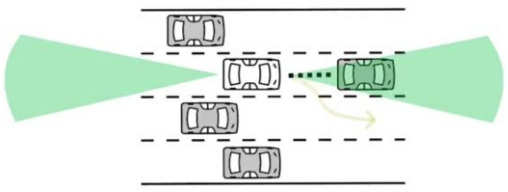
  - When merging, lateral coverage: usually current lane, adjacent lanes would be preferred



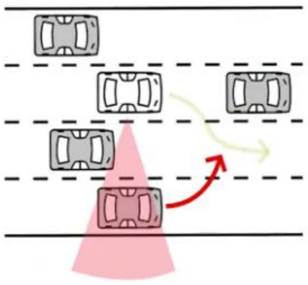
- Lane change:



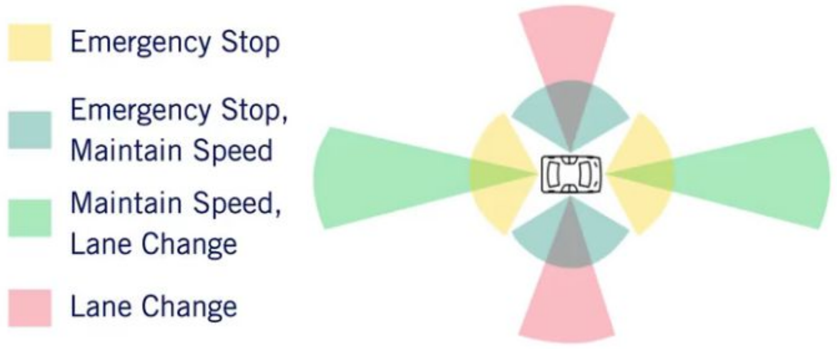
  - Longitude coverage: need to look forward to maintain a safe distance
  - Need to look behind to see what rear vehicles are doing



  - Lateral coverage: need wider sensing, e.g. what if a vehicle attempts to maneuver into the adjacent lane at the same time as we do.



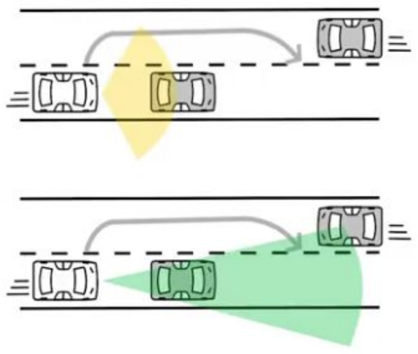
Overall



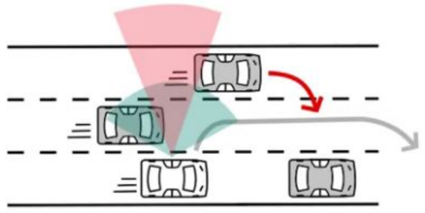
Urban pilot

6 kinds of maneuvers: emergency stop, maintain stop, lane change, overtaking, turning+crossing at intersections, passing roundabouts.
For the first three types of maneuvers, the coverage analysis is pretty much the same as the highway analysis but since we are not moving as quickly, we don't need the same extent for our **long-range sensing**.

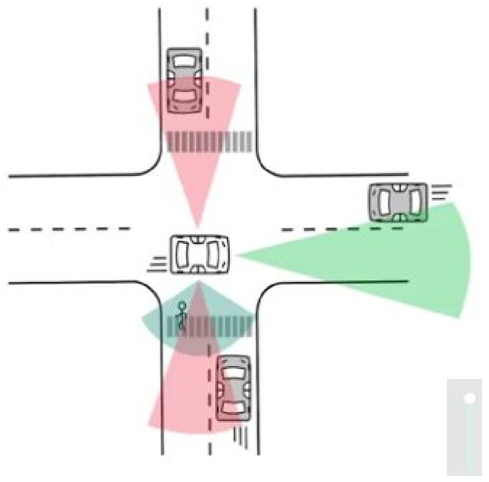
- Overtaking



Longitude coverage: if overtaking a parked or moving vehicle, need to detect incoming traffic beyond point of return to own lane.
Lateral coverage: always need to observe adjacent lanes. Need to observe additional lanes if other vehicles can move into adjacent lanes.

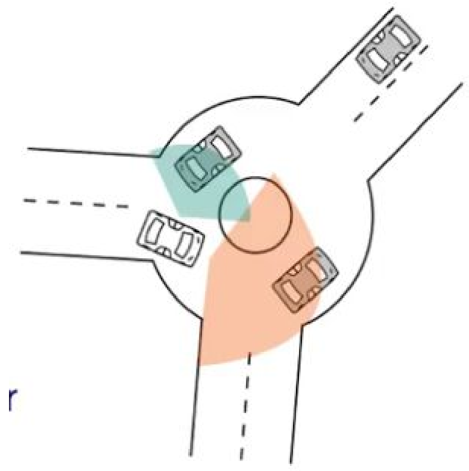


Intersections:
Observe beyond intersection for approaching vehicles, pedestrian crossing, clearing exit lanes
Requires near omni-directional sensing for arbitrary intersection angles.

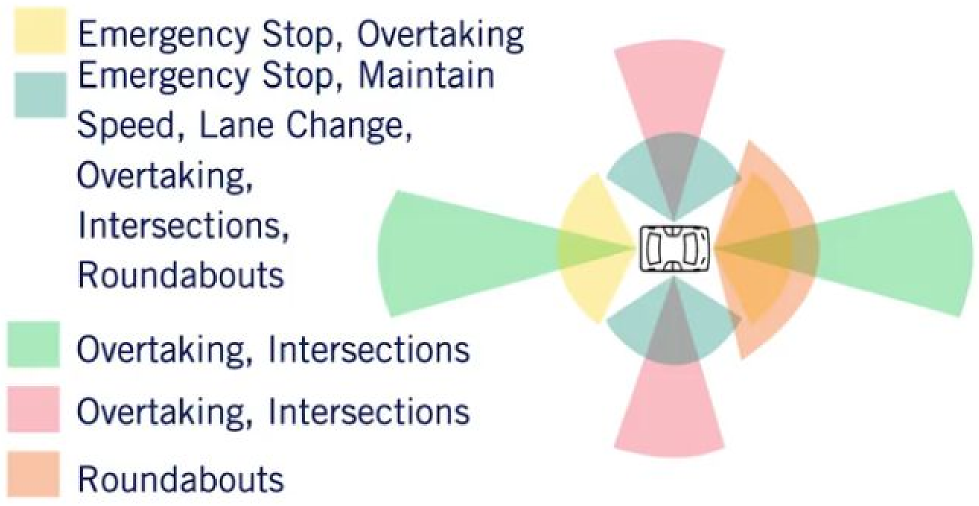


Roundabouts:

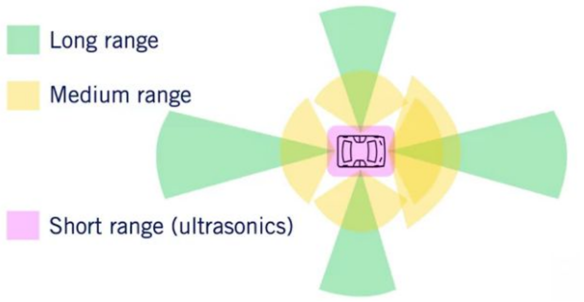
- Lateral range: vehicles are slower than usual, limited range requirements
- Longitudinal range: due to the shape of the roundabout, need a wider field of view.



Overall

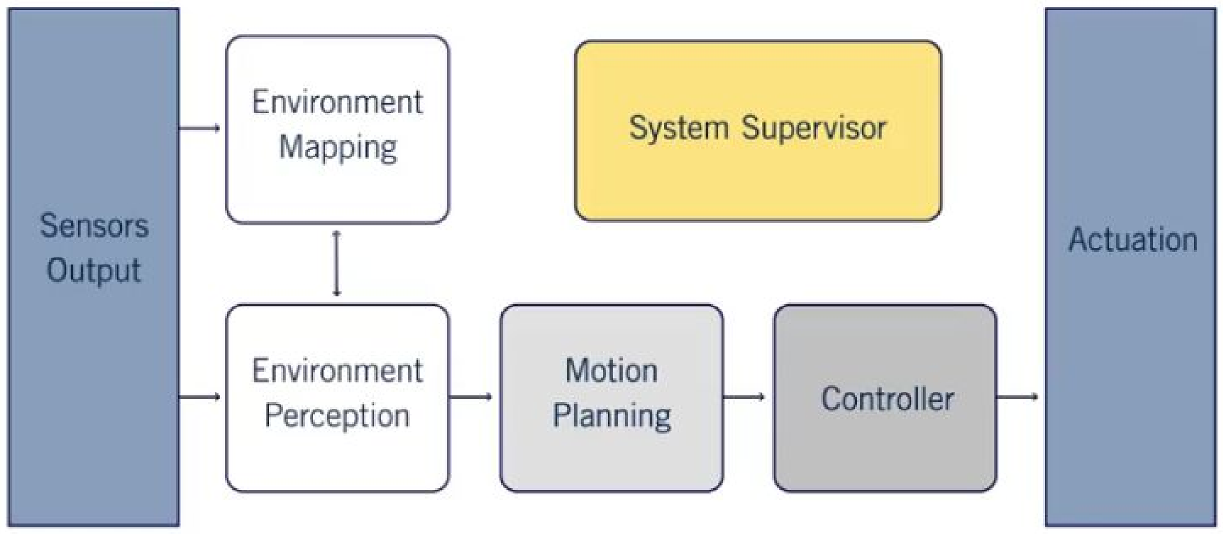

Emergency Stop, Overtaking
Emergency Stop, Maintain Speed, Lane Change, Overtaking, Intersections, Roundabouts
Overtaking, Intersections
Overtaking, Intersections
Roundabouts

Overall coverage and sensor analysis
We have a need for full 360 degrees sensor coverage on the short scale out to about 50 meters and much longer range requirements in the longitudinal direction. We can also add even shorter range sensors like sonar which are useful in parking scenarios.



Sensing requirements for an automated vehicle for highway and rural environments:
https://repository.tudelft.nl/islandora/object/uuid:2ae44ea2-e5e9-455c-8481-8284f8494e4e

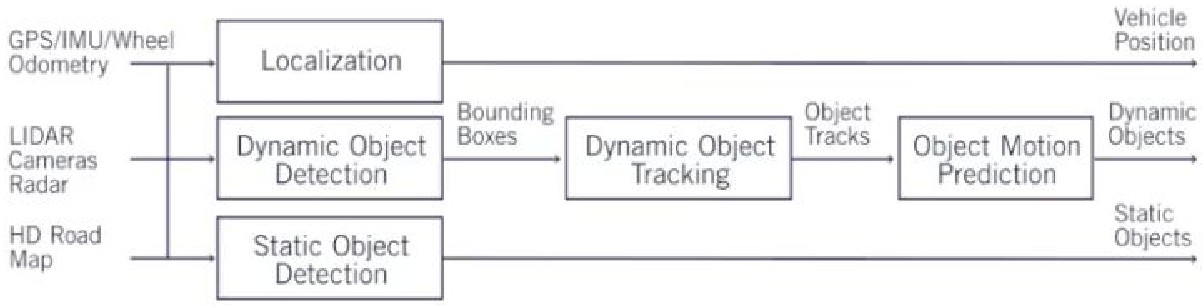
Software architecture:



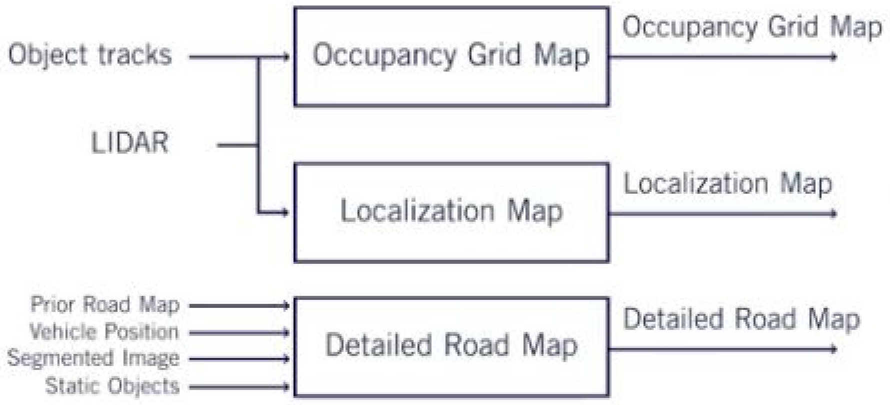
Modular Software Architecture

Standard software decomposition

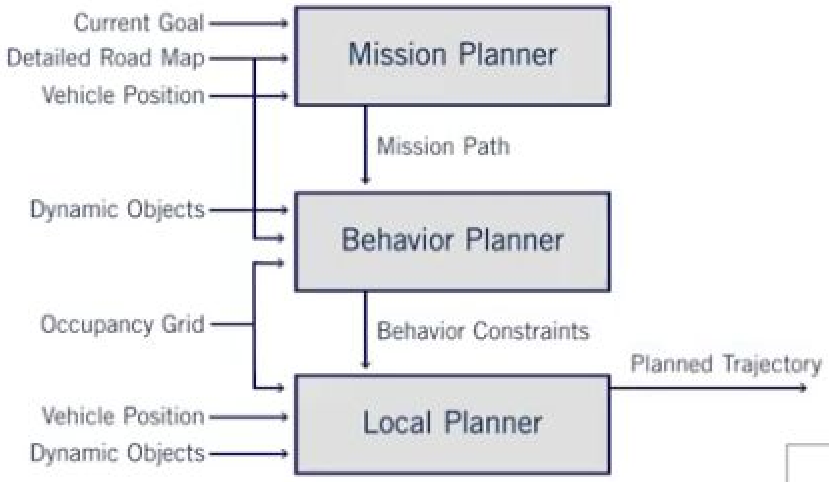
- Environment perception: Two key responsibilities, first, identifying the current location of the autonomous vehicle in space, and second, classifying and locating important elements of the environment for the driving task.



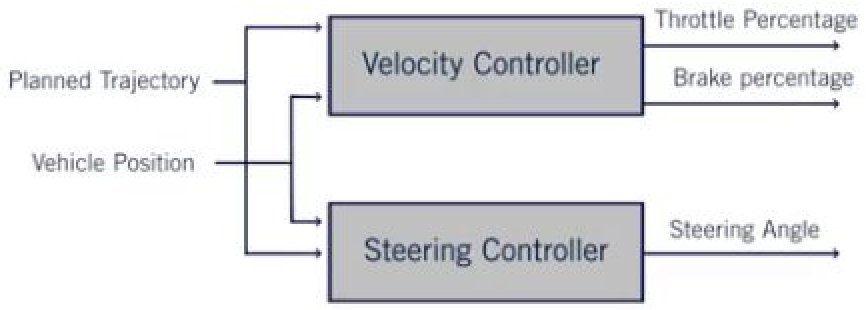
- Environment mapping: creates a set of maps which locate objects in the environment around the autonomous vehicle for a range of different uses, from collision avoidance to egomotion tracking and motion planning



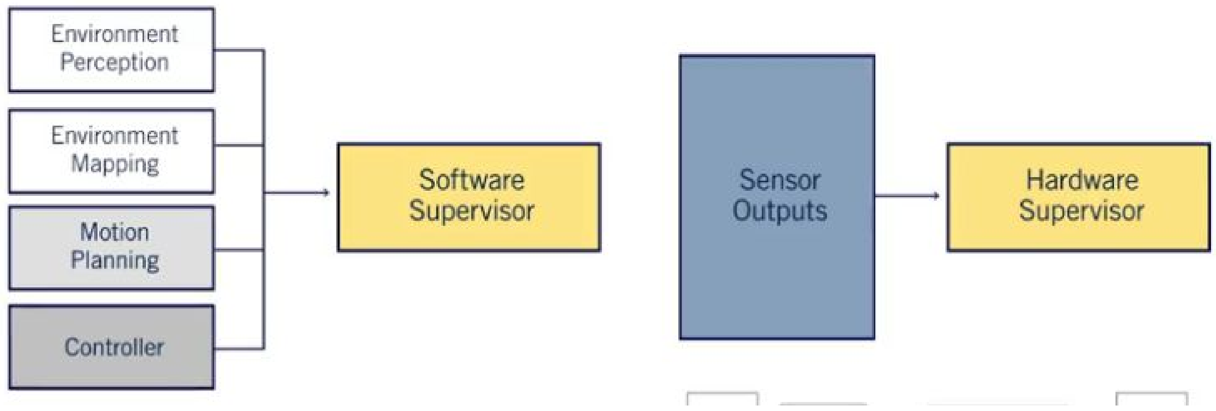
- Motion planning: The motion planning module makes all the decisions about what actions to take and where to drive based on all of the information provided by the perception and mapping modules. The motion planning module's main output is a safe, efficient and comfortable planned path that moves the vehicle towards its goal.



- Controller: The controller module takes the path and decides on the best steering angle, throttle position, brake pedal position, and gear settings to precisely follow the planned path.



- System supervisor: monitors all parts of the software stack, as well as the hardware output, to make sure that all systems are working as intended. It is also responsible for informing the safety driver of any problems found in the system.
    - Hardware supervisor: It continuously monitors all hardware components to check for any faults, such as a broken sensor, a missing measurement, or degraded information. Another responsibility of the hardware supervisor is to continuously analyze the hardware outputs for any outputs which do not match the domain which the self-driving car was programmed to perform under.
    - Software supervisor: It has the responsibility of validating this software stack to make sure that all elements are running as intended at the right frequencies and providing complete outputs. It also is responsible for analyzing inconsistencies between the outputs of all modules.



Environment maps

- Localization of vehicle in the environment
    - Localization point cloud or feature map.
      This map is created using a continuous set of lidar points or camera image features as the car moves through the environment. It is then used in

combination with GPS, IMU and wheel odometry by the localization module to accurately estimate the precise location of the vehicle at all times.
The difference between LiDAR maps is used to calculate the movement of the autonomous vehicle.

- Collision avoidance with static objects
    - Occupancy grid map
      Use LiDAR points to build a 2D/3D grid map which indicates the location of all static, or stationary, obstacles. This map is used to plan safe, collision-free paths for the autonomous vehicle.
- Path planning
    - Detailed road map: It contains detailed positions for all regulatory elements, regulatory attributes and lane markings.

Safety assurance for AD

Examples of AD crashes: Waymo, Uber, GM cruise,
Uber crash: multiple things gone wrong

- No realtime checks on safety drivers
- After the woman was detected on the road (6 sec before)
    - First classified as unknown object
    - Then misclassified as a vehicle
    - Then a bike
    - 1.3 sec before, Volvo system tried to do emergency braking maneurver

Uber crash report:
https://www.ntsb.gov/investigations/AccidentReports/Reports/HWY18MH010-prelim.pdf

Term

Harm to refer to the physical harm to a living thing.
Risk to describe the probability that an event occurs, combined with the severity of the harm, that the event can cause.
Safety: absence of unreasonable risk of harm
Hazard: potential source of unreasonable risk of harm

Major hazard sources

- Mechanical, like incorrect assembly of a brake system causing a premature failure
- Electrical: like internal wiring leading to a loss of indicator lighting.
- Hardware: chip errors
- Software: bugs
- Sensors: bad or noisy sensor data or inaccurate perception
- Behavior: planning or decision-making, or because the behavior selection for a specific scenario wasn't designed correctly.
- Fallback: the fallback to a human driver fails by not providing enough warning to the driver to resume responsibility or

- Cybersecurity: maybe a self-driving car gets hacked by some malicious entity.

NHTSA: safety framework

The National Highway Traffic Safety Administration: suggested area that OEMs should work on, not mandatory yet. It is a guideline.

System engineering approach to safety

Well-planned and controlled software development processes are essential, and the application of existing SAE and ISO standards from automotive, aerospace, and other relevant industries should be applied where relevant.
Autonomy design: which requires certain components to be included and considered in the autonomy software stack,

- Well defined ODD
- Well tested OEDR
- Reliable and convenient fallback
- Follow traffic laws and obeyed within ODD
- Avoid Cybersecurity threats
- HMI: Well convey the status of the machine at any point in time to the passenger / driver

Testing & Crash mitigation: which covers approaches to testing the autonomy functions and ways to reduce the negative effects of failures, as well as learning from them.

- Strong and extensive testing program: simulation, close track testing, and public road driving
- Crashworthiness: Crashes remain a reality of public road driving and autonomy systems that can minimize crash energy and exceed passenger safety standards in terms of restraints, airbags, and crash worthiness should be the norm.
- Post crash: The car must be rapidly returned to a safe state, for example, brought to a stop with fuel pumps securing the fuel, first responders alerted, and so on.
- Data recording: Black box. It is very helpful to have this crash data to analyze and design systems that can avoid the specific kind of crash in the future, and to resolve questions about what went wrong, and who was at fault during the event.
- Customer education: sufficient training

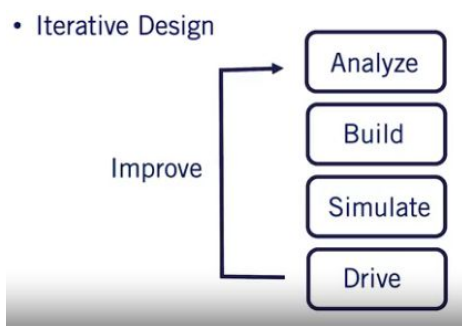
Waymo safety levels

Behavioral safety: This includes decisions that follow the traffic rules, can handle a wide range of scenarios within the ODD, and maintain vehicle safety through it.
Functional safety: Waymo ensures that the systems have backups and redundancies
Operational Safety: Its interfaces are usable and convenient and intuitive.
Crash safety: It designs systems that ensure minimum damage to people inside the car in the event of a crash.
Non-collision safety: This refers to system designs that minimize the danger to people that may interact with the system in some way, first responders, mechanics, hardware engineers, and so on

Waymo Safety processes

- Identify hazard scenarios and potential mitigations
- Use hazard assessment methods to define safety requirements
    - Preliminary analysis
    - Fault tree
    - Design failure mode & effects analysis
- Conduct extensive testing to make sure safety requirements are met

GM Cruise safety



Control car production
While Waymo relies on OEMs to design its vehicles and only discusses mechanical and electrical hazards related to its autonomy hardware, GM manufactures their cars entirely themselves and so can enforce a more integrated design with consistent quality standards throughout the self-driving hardware.
Safety through comprehensive risk management and deep integration

- Identify and address risk, validate solutions
- Prioritize elimination of risks, not just mitigation

All hardware, software systems meet

- Self-set standard for performance, crash protection, reliability, serviceability, security and safety

Safety processes

- Deductive analysis: Fault tree analysis
- Inductive analysis: design & process FMEA
- Explorative analysis: hazard and operability study

GM two major safety thresholds:

- Fail safety: the redundant functionality ensures the vehicle stops normally when the primary system fails
- SOTIF: all critical functionalities are evaluated for unpredictable scenarios

GM: testing

- Performance testing at different levels
- Requirements validation of components, levels
- Fault injection testing of safety critical functionality

- Intrusive testing: such as electromagnetic interference, etc.
- Durability testing and simulation based testing

Analytical safety

Ensuring the system works in theory and meets safety requirements found by hazard assessment. Namely, the system can be analyzed to define quantifiable safety performance or failure rates based on critical assessment of hazards and scenarios. In the end, analytic methods can only provide guidance on the safety performance of the self-driving system,

Data driven safety

Safety guarantee due to the fact that the system has performed autonomously without fail on the roads for a very large number of kms
In the US, on average, 1 fatal collision per 146 M km, 1 injury collision per 2.1 M km, ~1 collision per 400,000 km
https://safety.fhwa.dot.gov/rsdp/ddsa.aspx

Question

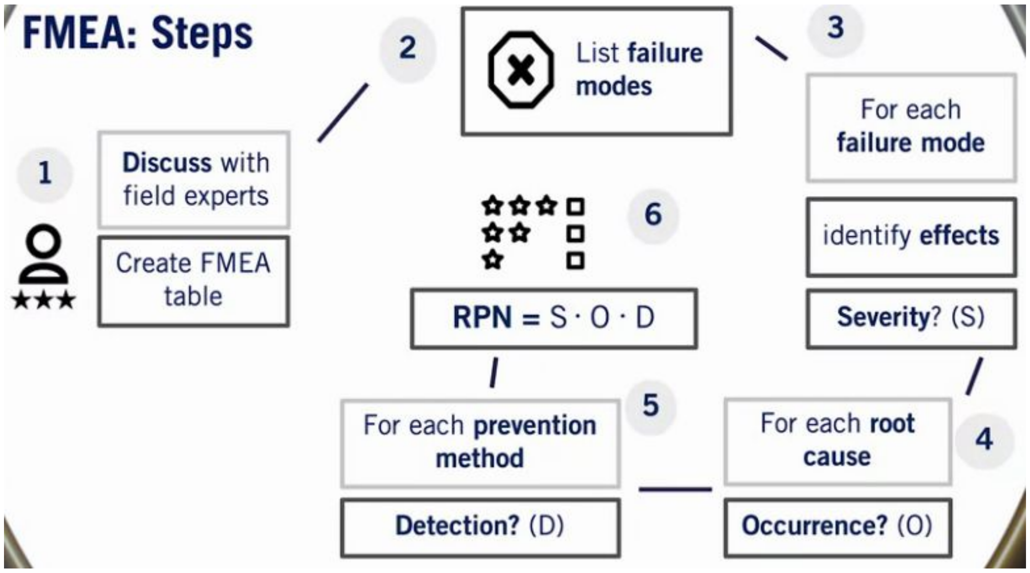
How many miles (years) would AVs have to drive to demonstrate with 95% confidence their failure rate to within 20% of the true rate of fatality per 146 M km?
~400 years, with a fleet of 100 vehicles traveling all the time.

Safety Frameworks for Self-Driving

Probabilistic Fault Tree Analysis
- Top down deductive failure analysis
- Boolean logic
- Assign probabilities to fault "leaves"
- Use logic gates to construct failure tree

Failure mode and effects analysis (FMEA)
- Bottom up process to identify all the effects of faults in a system

Failure mode: modes or ways in which a component of the system may fail
Effects analysis: analyzing the effects of the failure modes on the operation of the system
Categorize failure modes by priority
- How serious are their effects?
- How frequently do they happen?
- How easily can they be detected?



Risk Priority Number (RPN)

Hazard and operability study (HAZOP)

- Qualitative brainstorm process, needs "imagination"
- Use guide words to trigger brainstorming (such as not, more, less, etc)
- Applied to complex 'processes'
    - Sufficient design information is available and not likely to change significantly

Automotive Safety Framework

- ISO26262 Functional Safety Framework
- ISO/PAR 21448.1 - Safety of Intended Functionality

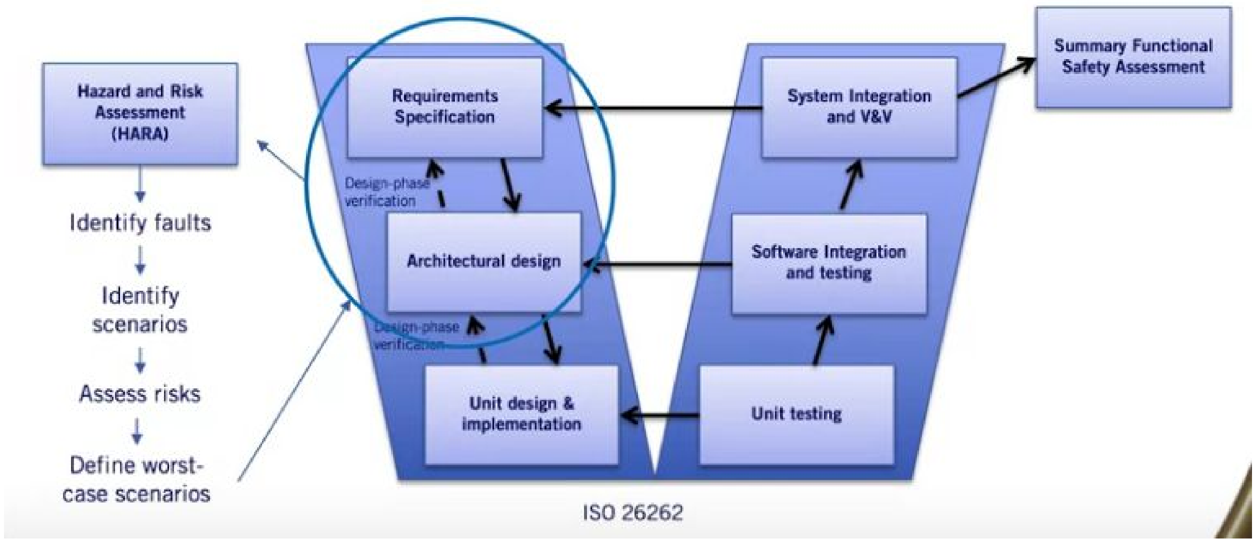
Functional Safety is defined as

- Safety due to absence of unreasonable risk
- Only concerned about malfunctioning system

ISO 26262 defines Automotive Safety Integrity Levels (ASILs)

ASIL-D most stringent, while ASIL-A least stringent

Functional Safety Process


Hazard and Risk Assessment (HARA)
Identify faults
Identify scenarios
Assess risks
Define worst-case scenarios
Requirements Specification
Design-phase verification
Architectural design
Design-phase verification
Unit design & implementation
Unit testing
Software Integration and testing
System Integration and V&V
Summary Functional Safety Assessment
ISO 26262

SOTIF

Failures due to performance limitations and misuse

- Sensor limitations
- Algorithm failures / insufficiencies
- User misuse - overload, confusion
- Designed for level 0-2 autonomy
- Can be seen as an extension of FuSa
    - V-shape process
    - Employ HARA

Kinematic Modeling in 2D

Generally, vehicle motion can be modeled either by considering the **geometric constraint that defines its motion** or by considering **all of the forces and moments acting on a vehicle**. The first case is known as Kinematic Modeling. Especially at low speeds when the accelerations are not significant, Kinematic Modeling is more than sufficient to capture the motion of a vehicle. When we instead include knowledge of the forces and moments acting on the vehicle, we're performing Dynamic Modeling. Dynamic models can do a great job of estimating vehicle motion throughout the vehicle's operating range, but are more involved to develop than kinematic models.

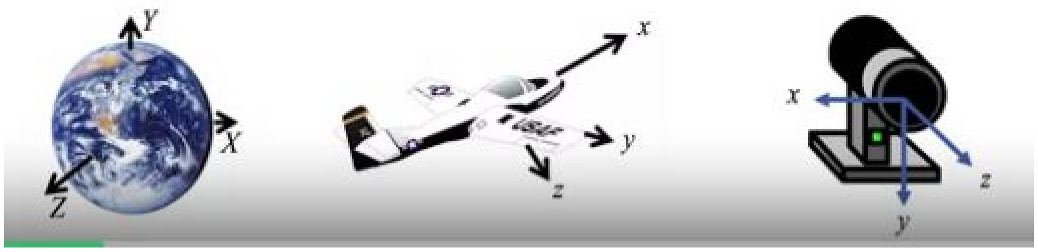
Kinematic vs dynamic modeling

- At low speeds, it is often sufficient to look only at kinematic models of vehicles

    - Examples: Two wheeled robot, bicycle model
- Dynamic model is more involved, but captures vehicle behavior more precisely over a wider operating range
    - Example: dynamic vehicle model

Coordinate frames

- Right handed by convention
- Inertial frame, global world system
    - Fixed usually relative to earth
    - Represented by East North Up (ENU) relative to a reference point nearby.
    - Or Earth-Centered Earth Fixed (ECEF), as is used in GNSS systems
- Body frame
    - Attached to vehicle, origin at vehicle center of gravity, or center of rotation
- Sensor frame
    - Attached to sensor, convenient for expressing sensor measurements



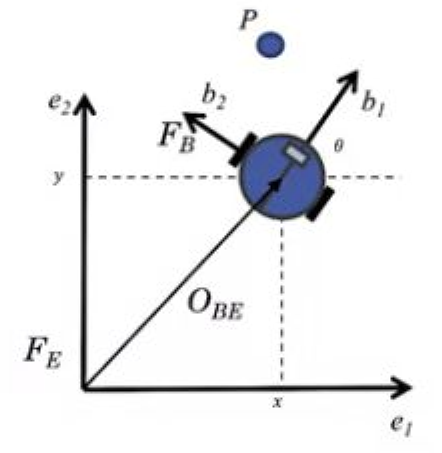
Coordinate transformation

- Conversion between inertial frame and body coordinates is done with a translation vector and a rotation matrix
    - Location of point (P) in body frame (B)

$$P_B = C_{EB}(\theta)\, P_E + O_{EB}$$

    - Location of point (P) in inertial frame (E)

$$P_E = C_{BE}(\theta)\, P_B + O_{BE}$$



Homogeneous Coordinate Form

- A 2D in homogeneous form

$$P = \begin{bmatrix} x \\ y \end{bmatrix} \quad \Longrightarrow \quad \bar{P} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Transforming a point from body to inertial coordinates with homogeneous coordinates

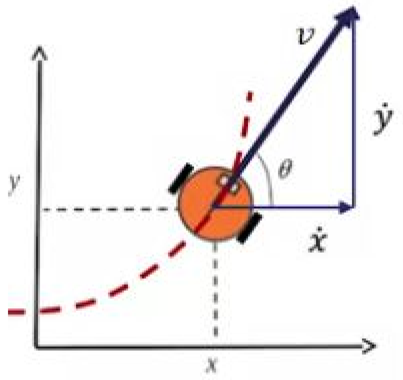
$$\overline{P_E} = [C_{EB}(\theta) \quad | \, O_{EB}]\overline{P_B}$$

2D kinematic modeling

The kinematic constraint is nonholonomic

- A constraint on rate of change of degrees of freedom
- Vehicle velocity always tangent to current path

$$\frac{dy}{dx} = \tan\theta = \frac{\sin\theta}{\cos\theta}$$



- Nonholonomic constraint

$$\dot{y}\cos\theta - \dot{x}\sin\theta = 0$$

- Velocity components

$$\dot{x} = v\cos\theta$$
$$\dot{y} = v\sin\theta$$

Simple robot motion kinematics

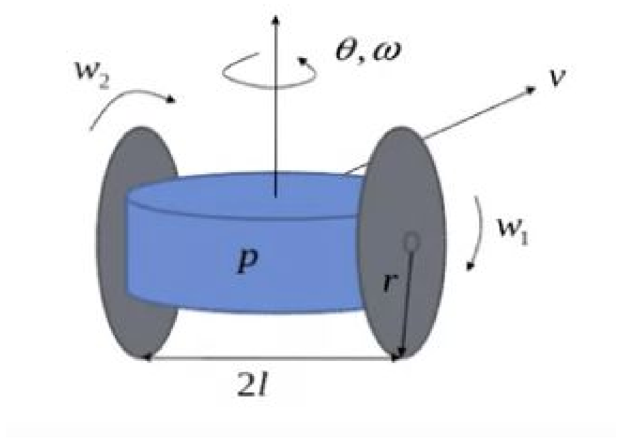
$$\begin{tikzcd}\boxed{\text{Inputs}} \arrow[r] & \boxed{\text{Simple Model}} \arrow[r] & \boxed{\text{States (Outputs)}}\end{tikzcd}$$

$$[v, \omega]$$

$$\dot{x} = v\cos\theta$$
$$\dot{y} = v\sin\theta$$
$$\dot{\theta} = \omega$$

A **state** is a set of variables often arranged in the form of a vector that fully describe the system at the current time.

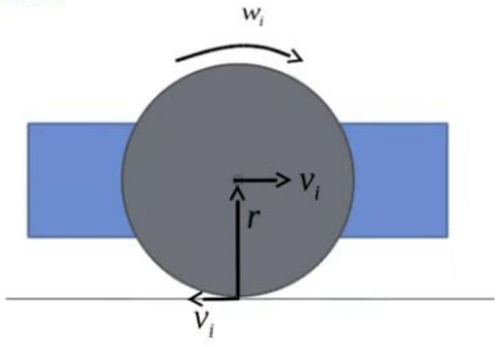
Two-wheeled robot kinematic model



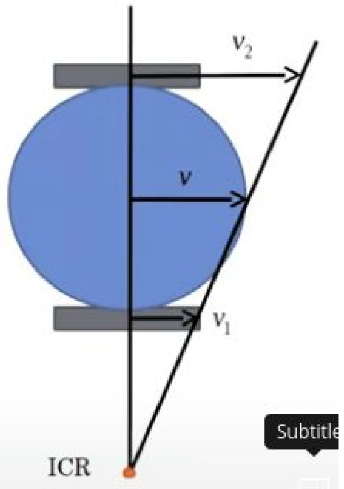
- Center: p
- Wheel to center: l
- Wheel radius: r
- Wheel rotation rates: w1, w2

- Kinematic constraint

$$v_i = rw_i$$



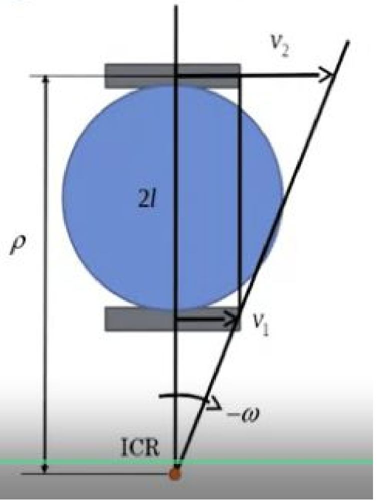
$$v = \frac{v_1 + v_2}{2} = \frac{rw_1 + rw_2}{2}$$

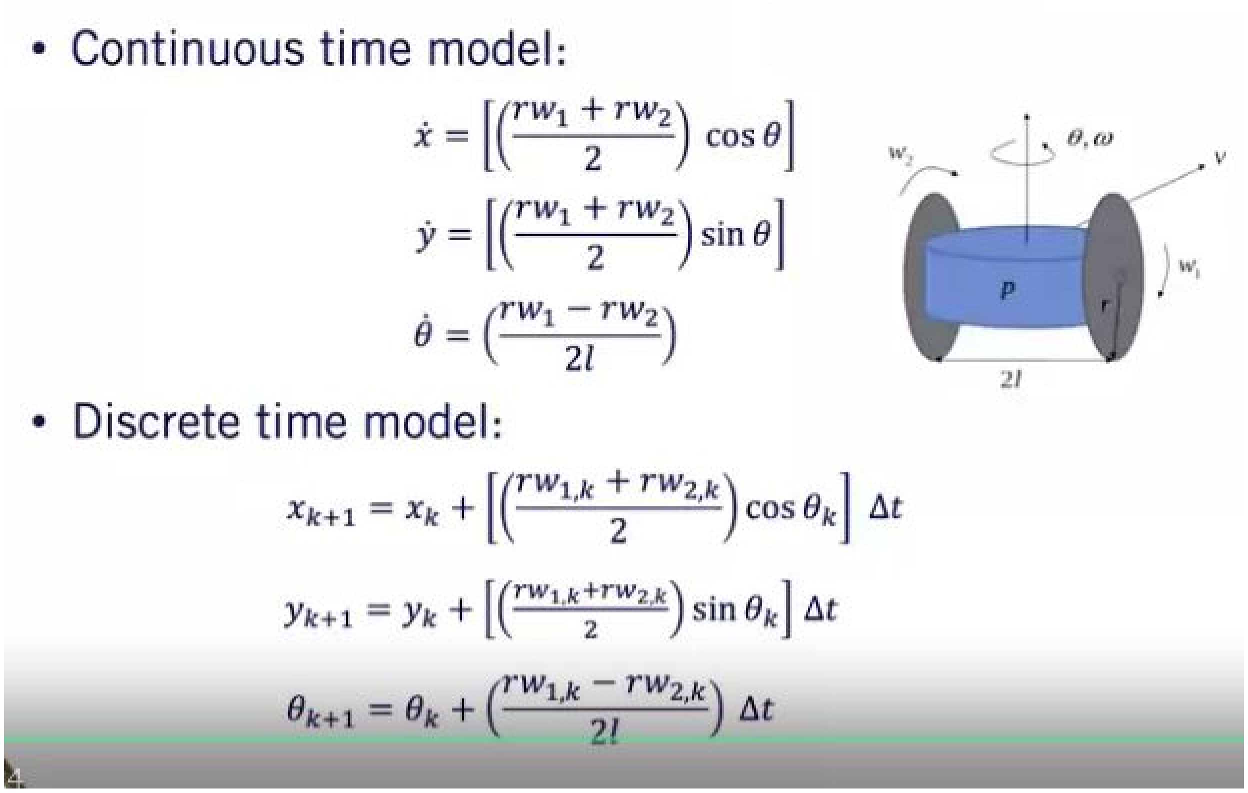


- Use the instantaneous center of rotation (ICR)
- Equivalent triangles give the angular rate of rotation

$$\omega = \frac{-v_2}{\rho} = \frac{-(v_2 - v_1)}{2l}$$

$$\omega = \frac{rw_1 - rw_2}{2l}$$





- Continuous time model:

$$\dot{x} = \left[\left(\frac{rw_1 + rw_2}{2}\right)\cos\theta\right]$$

$$\dot{y} = \left[\left(\frac{rw_1 + rw_2}{2}\right)\sin\theta\right]$$

$$\dot{\theta} = \left(\frac{rw_1 - rw_2}{2l}\right)$$

- Discrete time model:

$$x_{k+1} = x_k + \left[\left(\frac{rw_{1,k} + rw_{2,k}}{2}\right)\cos\theta_k\right]\Delta t$$

$$y_{k+1} = y_k + \left[\left(\frac{rw_{1,k} + rw_{2,k}}{2}\right)\sin\theta_k\right]\Delta t$$

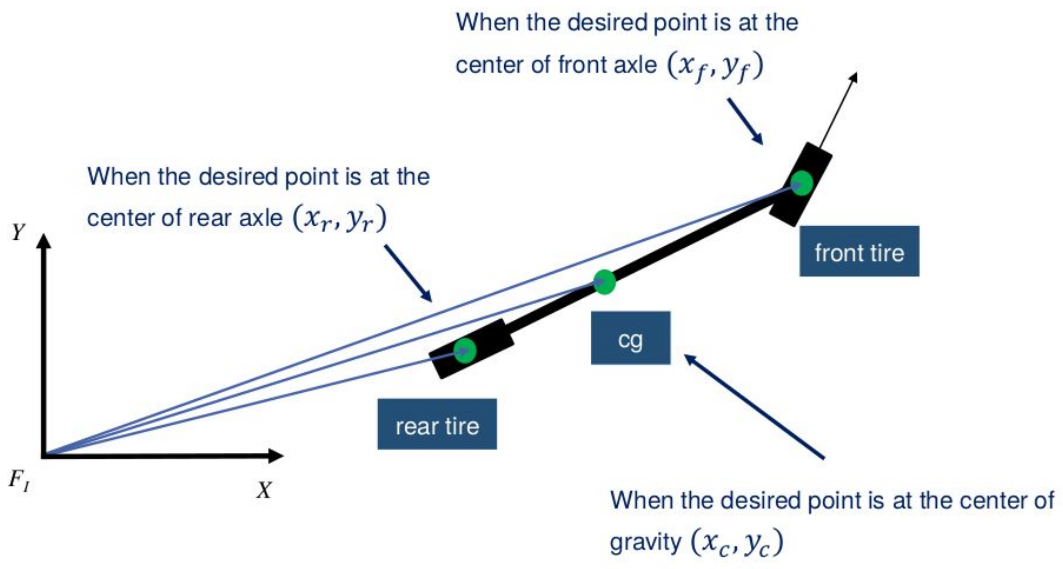
$$\theta_{k+1} = \theta_k + \left(\frac{rw_{1,k} - rw_{2,k}}{2l}\right)\Delta t$$

4

Bicycle kinematic model

The well-known **kinematic bicycle model** has long been used as a suitable control-oriented model for representing vehicles because of its simplicity and adherence to the nonholonomic constraints of a car.

- Front wheel steering


When the desired point is at the center of front axle (x_f, y_f)
When the desired point is at the center of rear axle (x_r, y_r)
Y
front tire
cg
rear tire
F_I
X
When the desired point is at the center of gravity (x_c, y_c)

- Rear wheel reference point

- Rear Wheel Reference Point
  - Apply Instantaneous Center of Rotation (ICR)

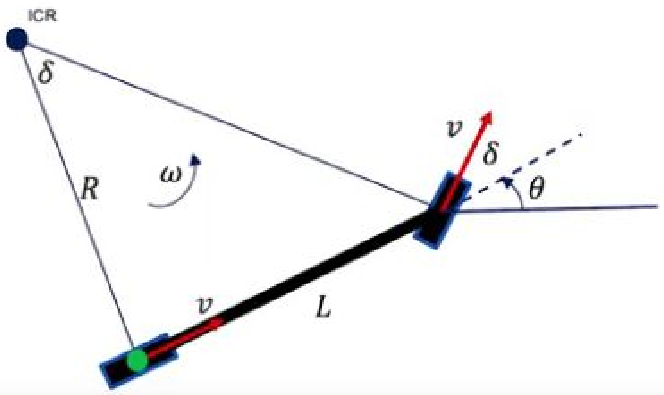
$$\dot{\theta} = \omega = \frac{v}{R}$$

  - Similar triangles

$$\tan\delta = \frac{L}{R}$$

  - Rotation rate equation

$$\dot{\theta} = \omega = \frac{v}{R} = \frac{v\tan\delta}{L}$$



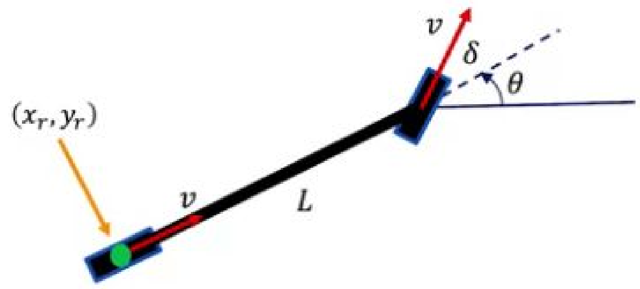
Rear Axle Bicycle Model

- If the desired point is at the center of the rear axle

$$\dot{x}_r = v\cos\theta$$

$$\dot{y}_r = v\sin\theta$$

$$\dot{\theta} = \frac{v\tan\delta}{L}$$

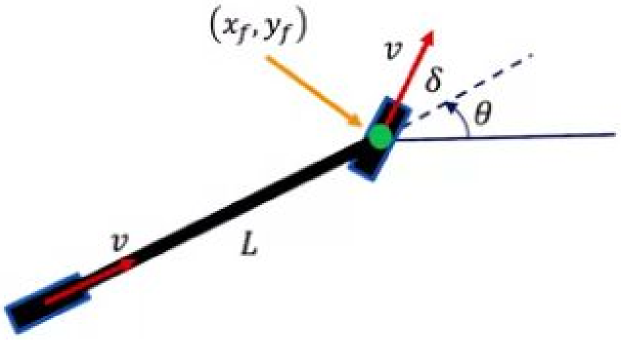


- If the desired point is at the center of the front axle

$$\dot{x}_f = v\cos(\theta + \delta)$$

$$\dot{y}_f = v\sin(\theta + \delta)$$

$$\dot{\theta} = \frac{v\sin\delta}{L}$$



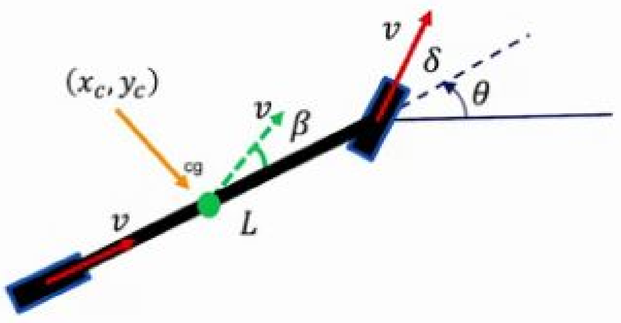
- If the desired point is at the center of the gravity (cg)

$$\dot{x}_c = v\cos(\theta + \beta)$$

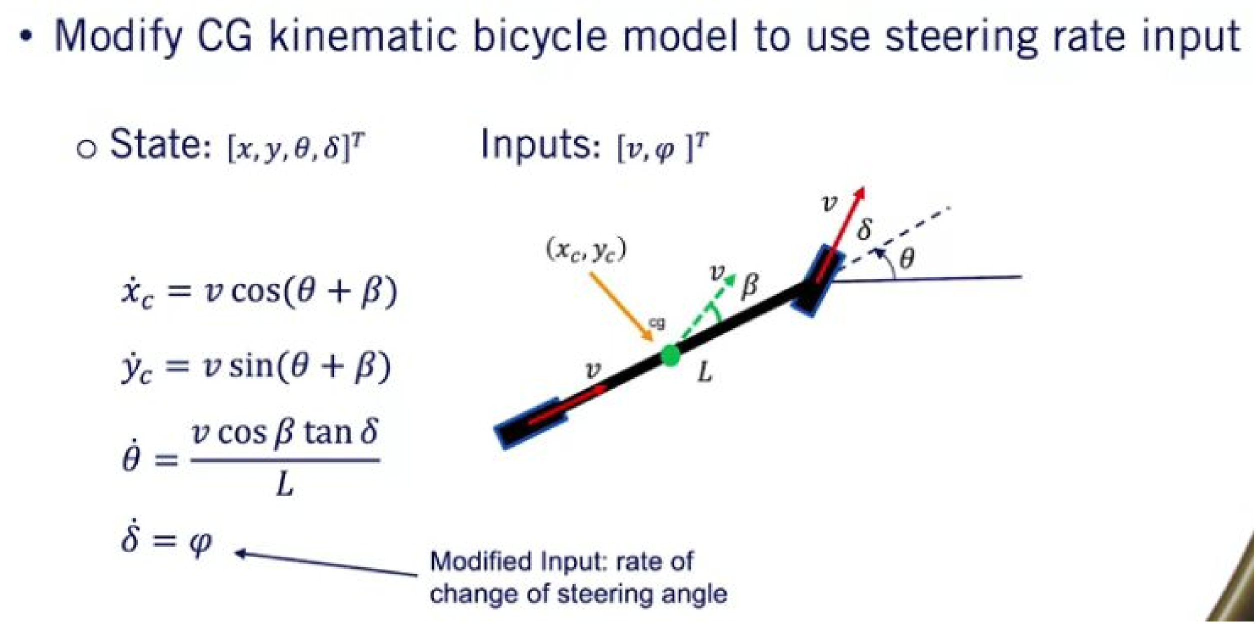
$$\dot{y}_c = v\sin(\theta + \beta)$$

$$\dot{\theta} = \frac{v\cos\beta\tan\delta}{L}$$

$$\beta = \tan^{-1}\left(\frac{l_r\tan\delta}{L}\right)$$



State space representation



- Modify CG kinematic bicycle model to use steering rate input
  - State: $[x, y, \theta, \delta]^T$ Inputs: $[v, \varphi]^T$

$$\dot{x}_c = v\cos(\theta + \beta)$$

$$\dot{y}_c = v\sin(\theta + \beta)$$

$$\dot{\theta} = \frac{v\cos\beta\tan\delta}{L}$$

$$\dot{\delta} = \varphi$$

Modified Input: rate of change of steering angle

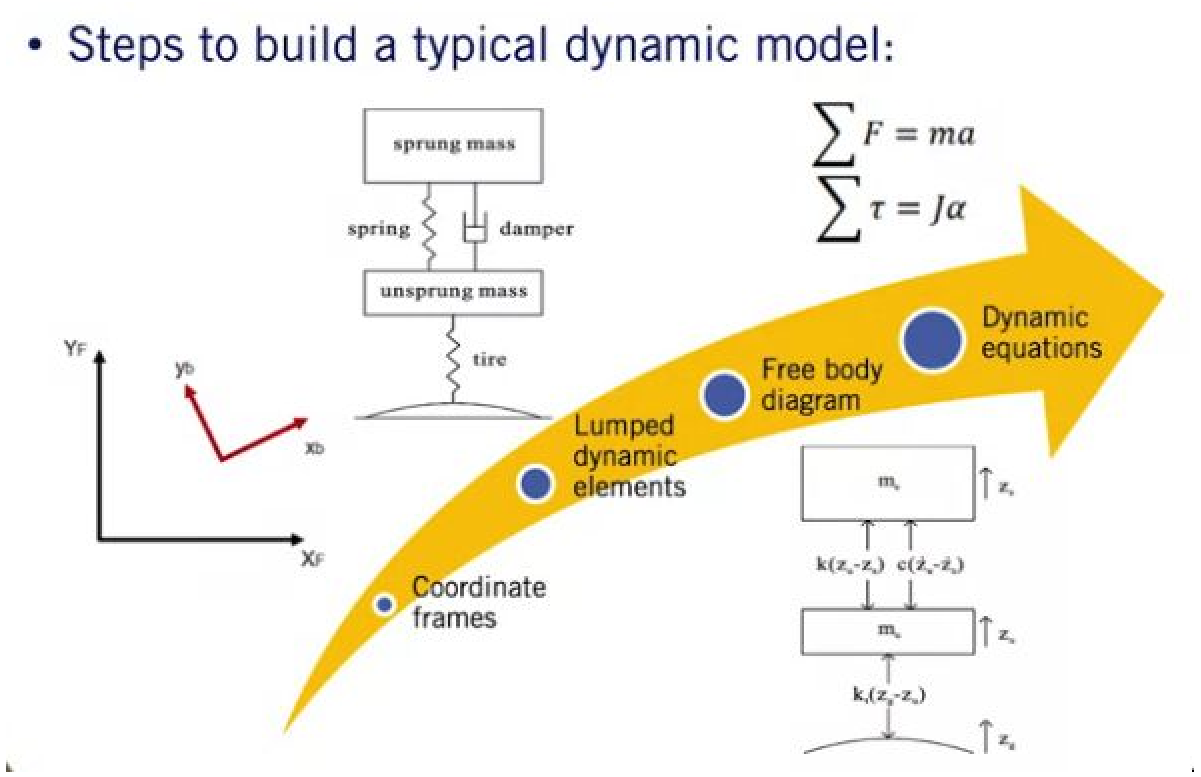
Dynamic Modeling in 2D

Why is Dynamic Modeling important?

- At higher speed and slippery road, vehicles do not satisfy no slip condition
- Forces such as drag, road friction govern require throttle input

Steps to build a typical DM

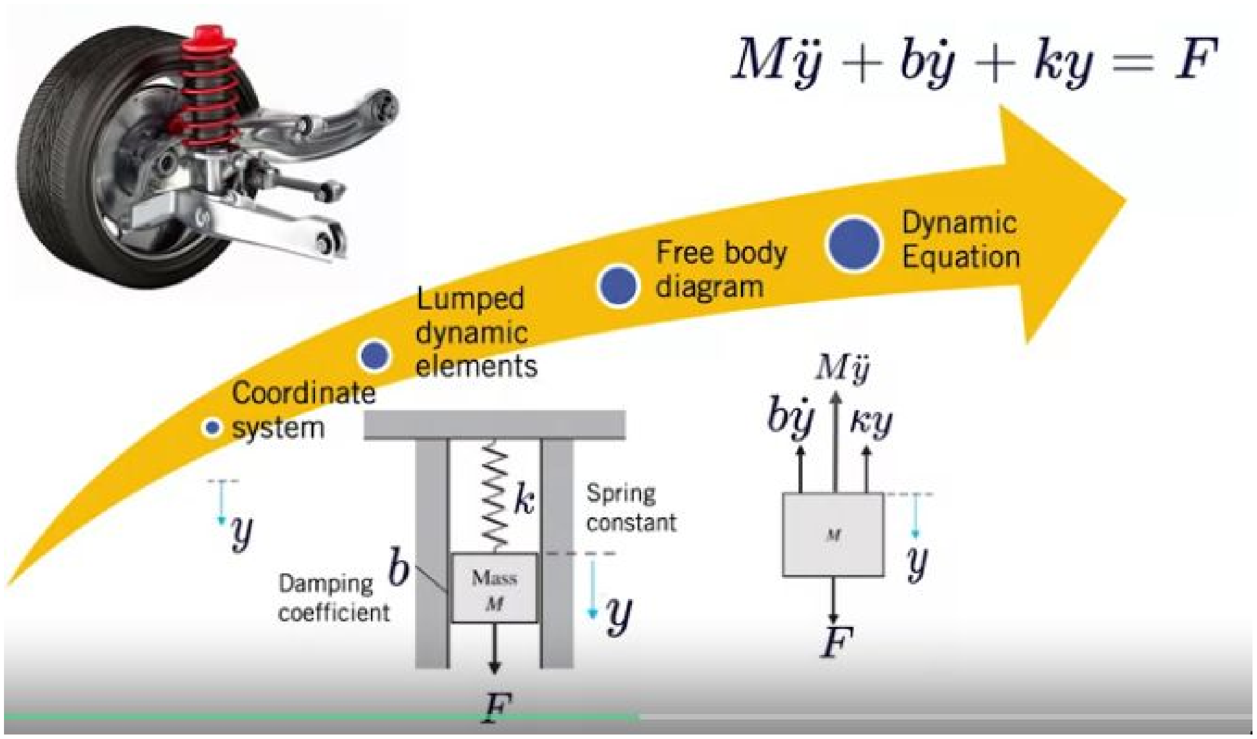
- Coordinate frames
- Lumped dynamic elements
- Free body diagram
- Dynamic equations



Translation system

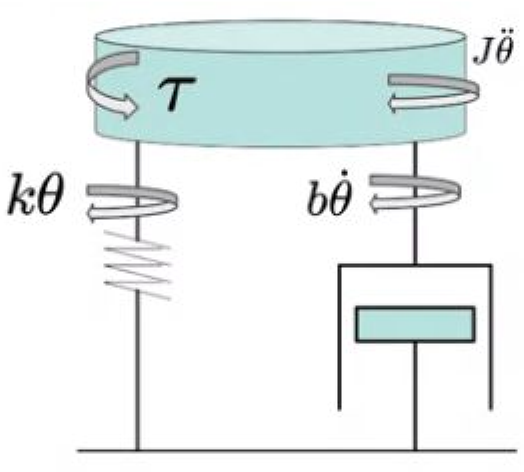
- Deals with forces and torques
- Need to equate all forces
- Governed by Newton’s second law

Example

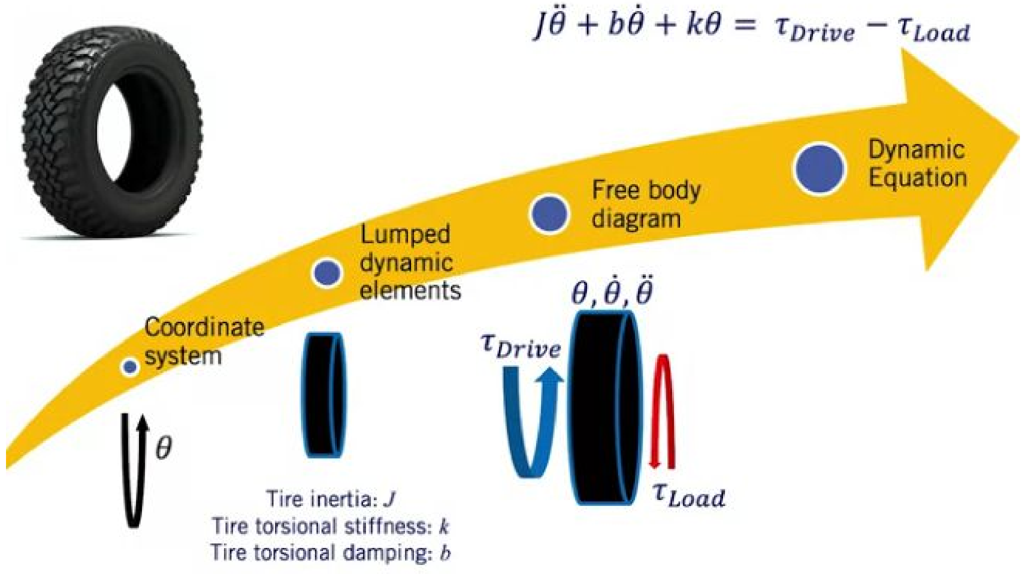


Rotational system

- Inertial J
- Torsional force T
- Forces resisting that torsional force
    - Sprint force
    - Damping force
    - Inertia force



Example

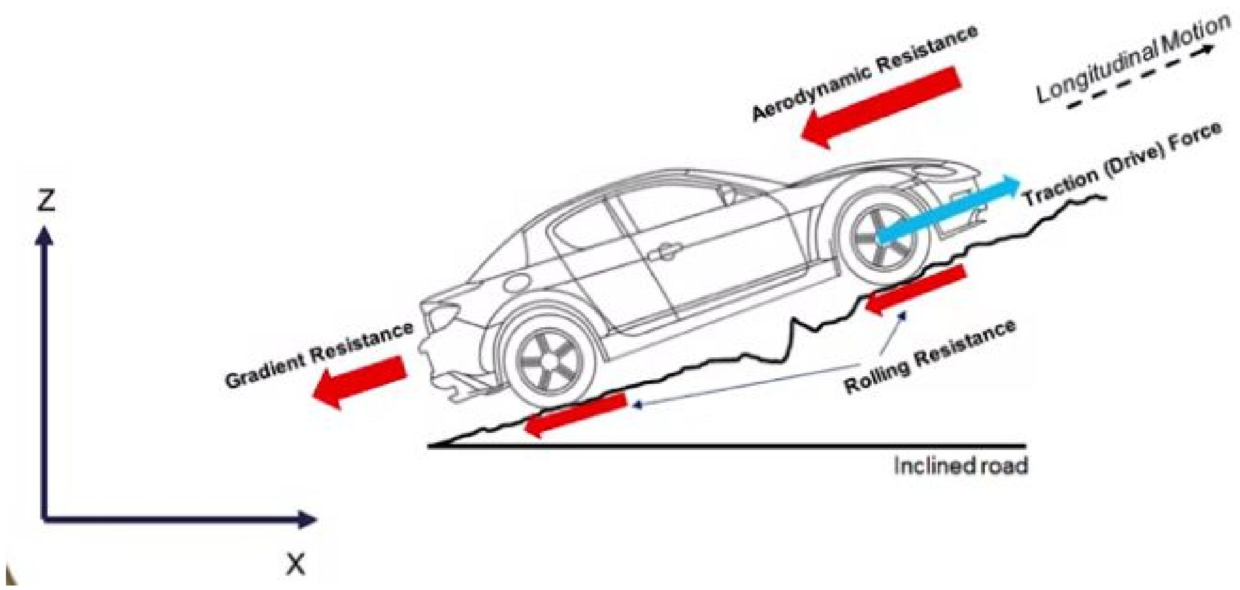

$J\ddot{\theta} + b\dot{\theta} + k\theta = \tau_{Drive} - \tau_{Load}$
Coordinate system
Lumped dynamic elements
Free body diagram
Dynamic Equation
θ
Tire inertia: J
Tire torsional stiffness: k
Tire torsional damping: b
$\theta, \dot{\theta}, \ddot{\theta}$
τ_{Drive}
τ_{Load}

Applications

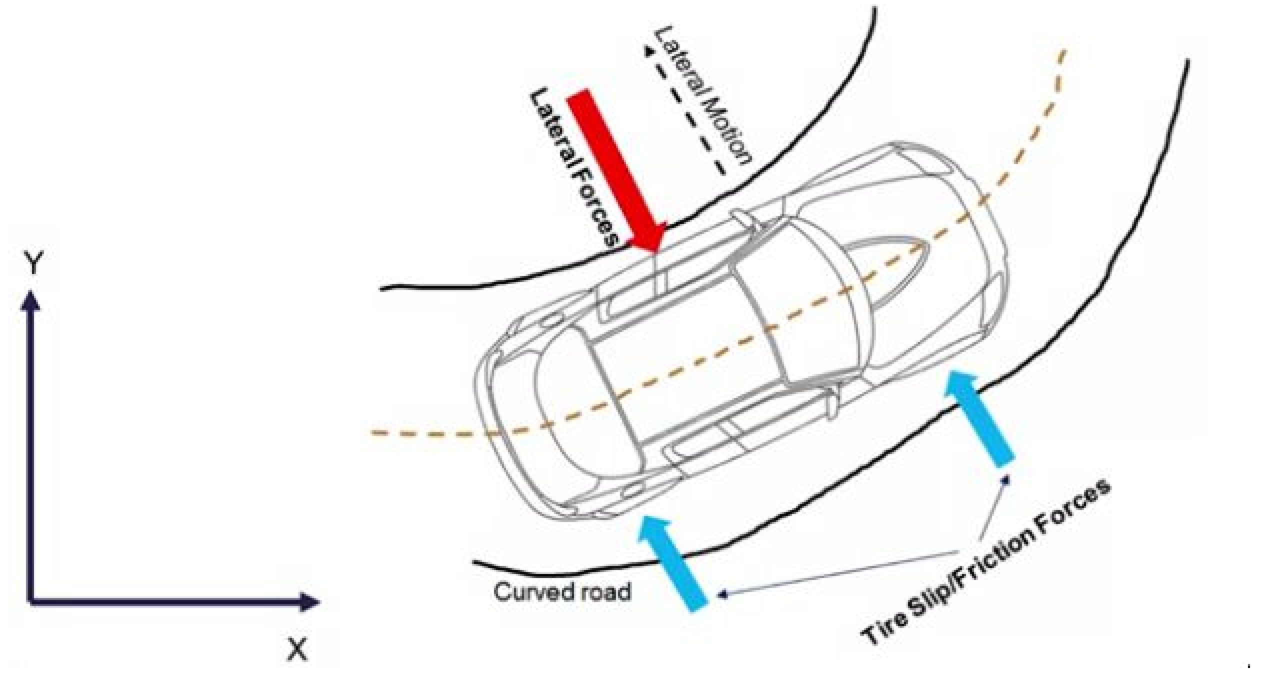
All components, forces and moments in 3D

- Pitch, roll, normal forces
- Suspension, drivetrain, component models

Vehicle longitudinal motion



Vehicle lateral motion



General Dynamics:

Ardema, Mark D. *Newton-Euler Dynamics*, Springer: Santa Clara University, Santa Clara (2005).

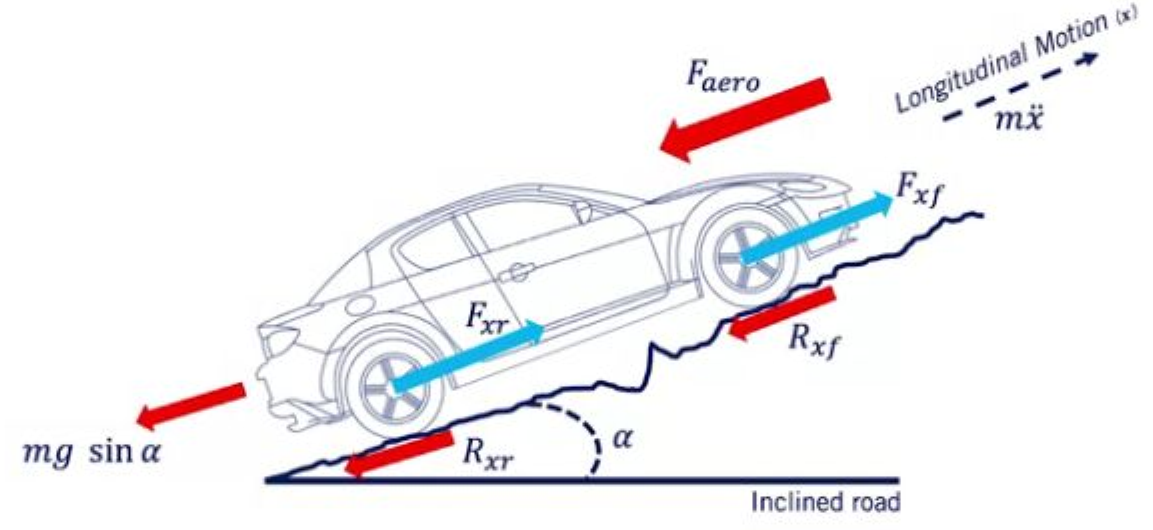
Tong, David. *Classical Dynamics* University of Cambridge Course Notes (2004)

Vehicle Modeling:

Rajamani, Rajesh. *Vehicle dynamics and control,* Springer Science & Business Media (2011).

Jacobson, Bengt, et al. *Vehicle Dynamics,* Vehicle Dynamics Group, Division of Vehicle and Autonomous Systems, Department of Applied Mechanics, Chalmers University of Technology (2016)

Longitudinal vehicle model



$$\overbrace{m\ddot{x}}^{\text{Vehicle acceleration}} = \overbrace{F_{xf} + F_{xr}}^{\text{Front \& rear tire forces}} - \overbrace{F_{aero}}^{\text{Aerodynamic forces}} - \overbrace{R_{xf} - R_{xr}}^{\text{Front \& rear road rolling resistance}} - \overbrace{mg\sin\alpha}^{\text{Gravitational force due to the road inclination}}$$

Or simplified as

- Let F_x - total longitudinal force: $F_x = F_{xf} + F_{xr}$
- Let R_x - total rolling resistance: $R_x = R_{xf} + R_{xr}$
- Assume α is a small angle: $\sin\alpha = \alpha$

- Then the simplified longitudinal dynamics become

$$\underbrace{m\ddot{x}}_{\text{Inertial Term}} = \underbrace{F_x}_{\text{Traction Force}} - \underbrace{\boxed{F_{aero} - R_x - mg\alpha}}_{\text{Total Resistant Forces } (F_{Load})}$$

Simple resistance force models

- Total resistance load:

$$F_{load} = F_{aero} + R_x + mg\alpha$$

- The aerodynamic force can depend on air density, frontal area, on the speed of the vehicle

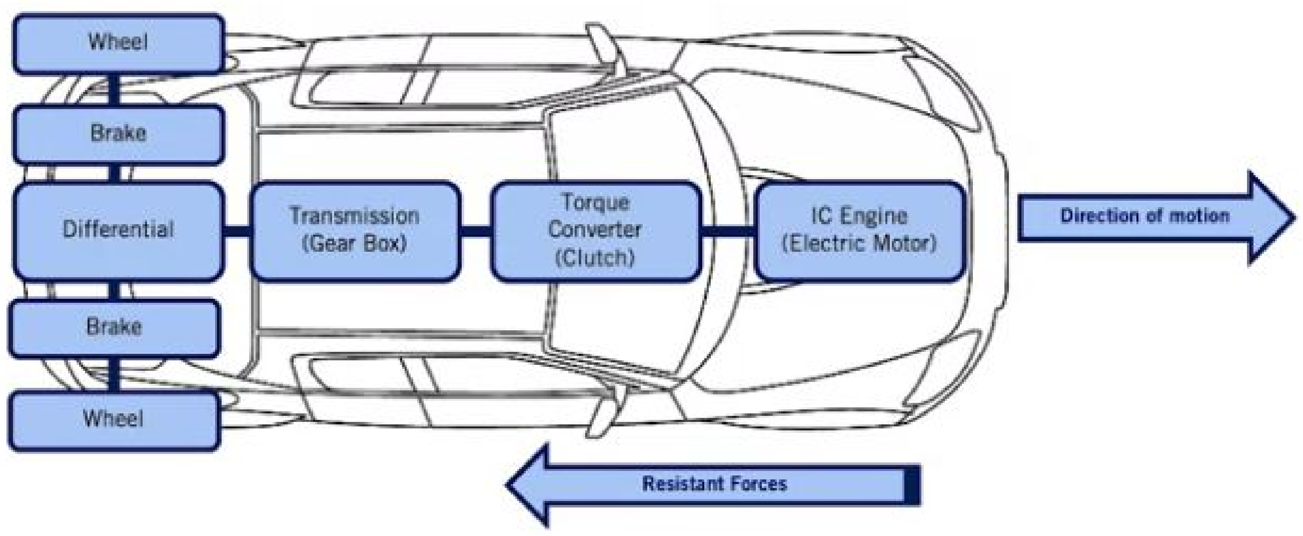
$$F_{aero} = {}^{1}/_{2}\, C_a \rho A \dot{x}^2 = \boxed{c_a} \dot{x}^2$$

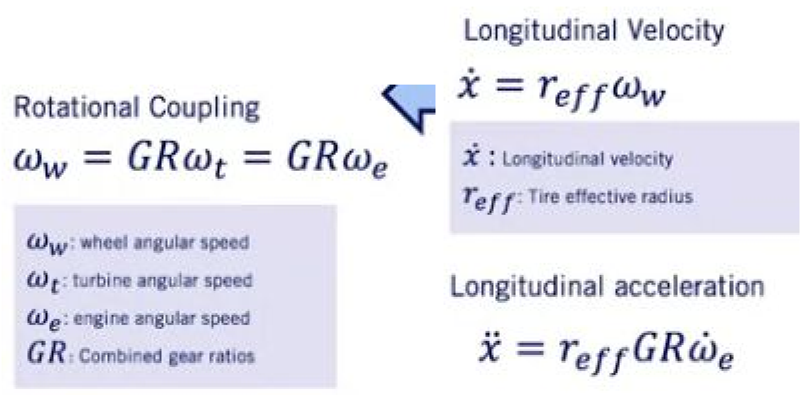
- The rolling resistance can depend on the tire normal force, tire pressures and vehicle speed

$$R_x = N(\hat{c}_{r,0} + \hat{c}_{r,1}|\dot{x}| + \hat{c}_{r,2}\dot{x}^2) \approx \boxed{c_{r,1}} |\dot{x}|$$

Powertrain modeling

The vehicle powertrain determines the vehicle's forward velocity and acceleration.





Rotational Coupling

$$\omega_w = GR\omega_t = GR\omega_e$$

ω_w: wheel angular speed

ω_t: turbine angular speed

ω_e: engine angular speed

GR: Combined gear ratios

Longitudinal Velocity

$$\dot{x} = r_{eff}\omega_w$$

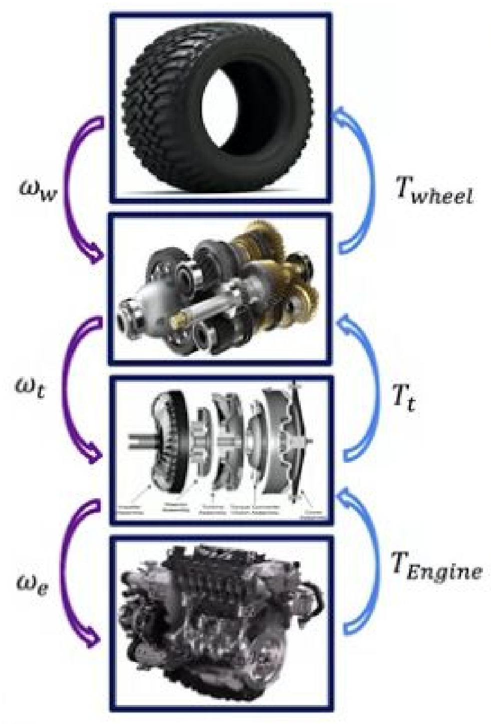
$\dot{x}$: Longitudinal velocity

r_{eff}: Tire effective radius

Longitudinal acceleration

$$\ddot{x} = r_{eff}GR\dot{\omega}_e$$

Power flow in powertrain



Wheel

$$I_w\dot{\omega}_w = T_{wheel} - r_{eff}F_x$$

$$T_{wheel} = I_w\dot{\omega}_w + r_{eff}F_x$$

Transmission

$$I_t\dot{\omega}_t = T_t - (GR)T_{wheel}$$

$$I_t\dot{\omega}_t = T_t - GR(I_w\dot{\omega}_w + r_{eff}F_x)$$

Torque Converter

$$\omega_t = \omega_e$$

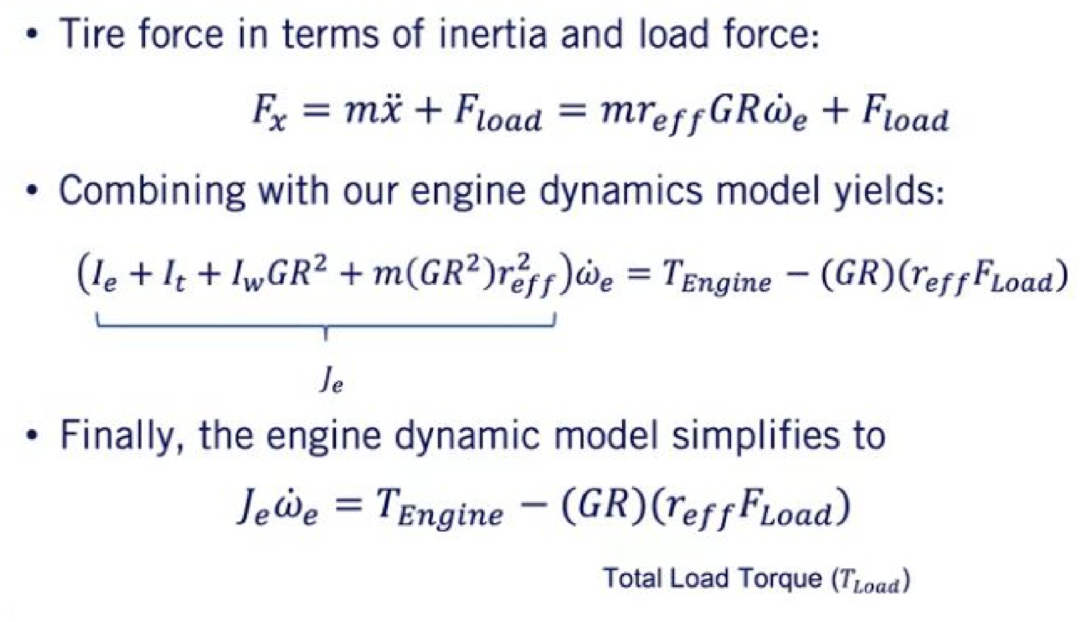
$$T_t = (I_t + I_wGR^2)\dot{\omega}_e + GRr_{eff}F_x$$

Engine

$$I_e\dot{\omega}_e = T_{Engine} - T_t$$

$$I_e\dot{\omega}_e = T_{Engine} - (I_t + I_wGR^2)\dot{\omega}_e - GRr_{eff}F_x$$

Engine dynamics



- Tire force in terms of inertia and load force:

$$F_x = m\ddot{x} + F_{load} = m r_{eff} GR \dot{\omega}_e + F_{load}$$

- Combining with our engine dynamics model yields:

$$\underbrace{\left(I_e + I_t + I_w GR^2 + m(GR^2) r_{eff}^2\right)}_{J_e} \dot{\omega}_e = T_{Engine} - (GR)(r_{eff} F_{Load})$$

- Finally, the engine dynamic model simplifies to

$$J_e \dot{\omega}_e = T_{Engine} - (GR)(r_{eff} F_{Load})$$

Total Load Torque (T_{Load})

Lateral Dynamics of Bicycle Model

Vehicle model to bicycle model

Assumptions:

- Long velocity is constant
- Left and right axle are lumped into a single wheel (bicycle model)
- Suspension movement, road inclination and aerodynamic influence are neglected

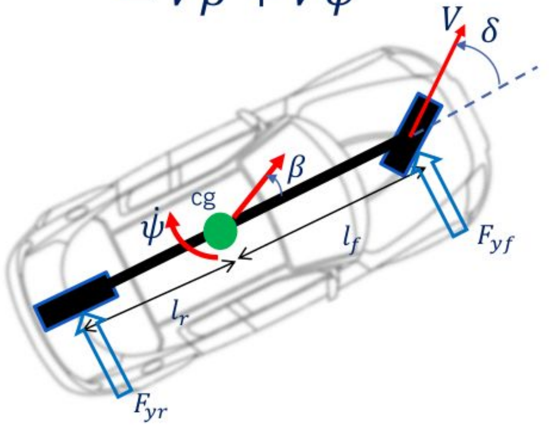
Lateral dynamics

We use the vehicle center of gravity as the reference point for the dynamic model as it simplifies the application of Newton’s second law.

- Lateral dynamics can be written as

Lateral acceleration

$$a_y = \ddot{y} + \omega^2 R$$
$$= V\dot{\beta} + V\dot{\psi}$$



vehicle mass

side slip rate

yaw rate

front and rear tire forces

$$mV(\dot{\beta} + \dot{\psi}) = F_{yf} + F_{yr}$$

vehicle velocity

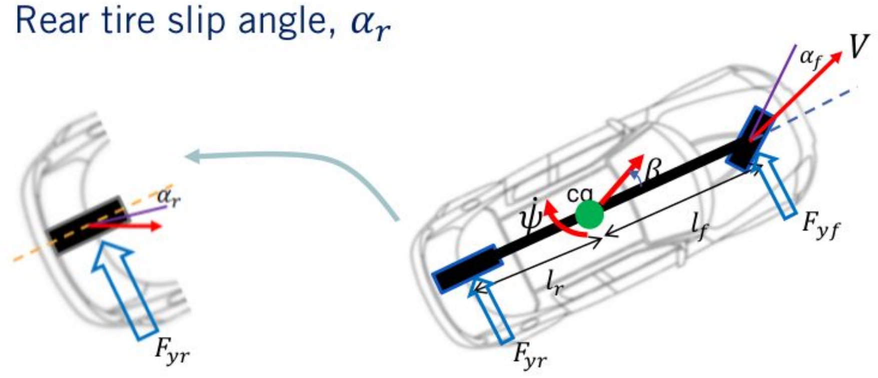
$$I_z\ddot{\psi} = l_f F_{yf} - l_r F_{yr}$$

vehicle inertia

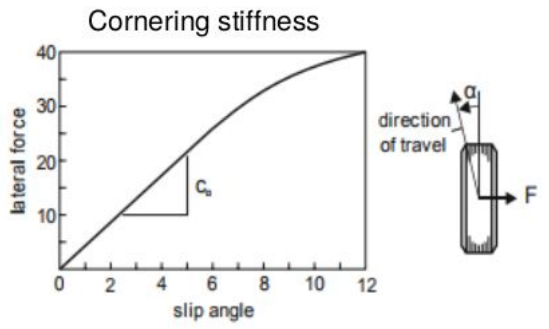
center of gravity distance from front and rear tires

Tire slip angles

- Many different tire slip models
- For small tire slip angles, the lateral tire forces are approximated as a linear function of tire slip angle
- Tire variables
  - Front tire slip angle, α_f
  - Rear tire slip angle, α_r



Front and rear tire forces



- C_f :linearized cornering stiffness of the front wheel

$$F_{yf} = C_f \alpha_f = C_f\left(\delta - \beta - \frac{l_f \dot{\psi}}{V}\right)$$

- C_r : linearized cornering stiffness of the rear wheel

$$F_{yr} = C_r \alpha_r = C_r\left(-\beta + \frac{l_r \dot{\psi}}{V}\right)$$

Lateral and yaw dynamics

- From the previous slide formulations:

$$F_{yf} = C_f \alpha_f = C_f\left(\delta - \beta - \frac{l_f \dot{\psi}}{V}\right)$$
$$F_{yr} = C_r \alpha_r = C_r\left(-\beta + \frac{l_r \dot{\psi}}{V}\right)$$

Substitute the lateral forces

$$mV(\dot{\beta} + \dot{\psi}) = F_{yf} + F_{yr}$$
$$I_z \ddot{\psi} = l_f F_{yf} - l_r F_{yr}$$

Rearranging the equations

$$\dot{\beta} = \frac{-(C_r + C_f)}{mV}\beta + \left(\frac{C_r l_r - C_f l_f}{mV^2} - 1\right)\dot{\psi} + \frac{C_f}{mV}\delta$$
$$\ddot{\psi} = \frac{C_r l_r - C_f l_f}{I_z}\beta - \frac{C_r l_r^2 + C_f l_f^2}{I_z V}\dot{\psi} + \frac{C_f l_f}{I_z}\delta$$

Standard state space representation

- State Vector: $X_{lat} = [\underset{\text{lateral position}}{y} \quad \underset{\text{side slip angle}}{\beta} \quad \underset{\text{yaw angle}}{\psi} \quad \underset{\text{yaw rate}}{\dot{\psi}}]^T$

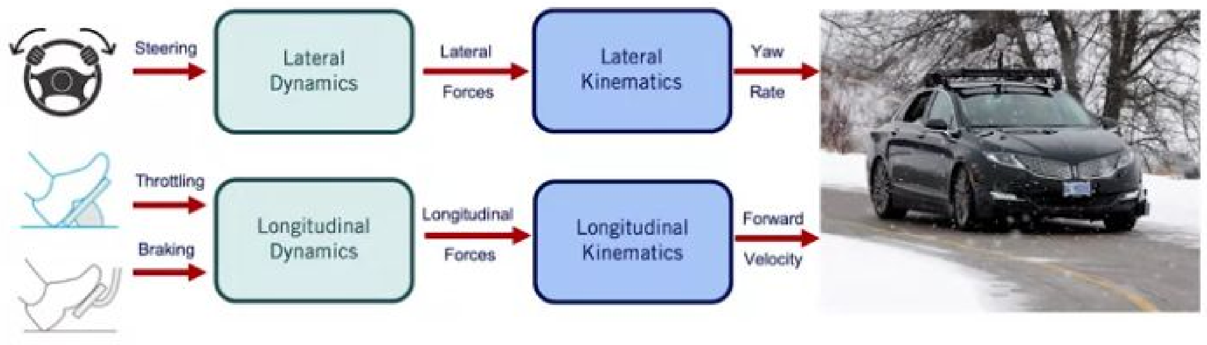
$$\boxed{\dot{X}_{lat} = A_{lat}X_{lat} + B_{lat}\delta}$$

$$A_{lat} = \begin{bmatrix} 0 & V & V & 0 \\ 0 & -\frac{C_r + C_f}{mV} & 0 & \frac{C_r l_r - C_f\, l_f}{mV^2} - 1 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{C_r l_r - C_f\, l_f}{I_z} & 0 & -\frac{C_r l_r^2 + C_f\, l_f^2}{I_z V} \end{bmatrix}$$

$$B_{lat} = \begin{bmatrix} 0 \\ \frac{C_f}{mV} \\ 0 \\ \frac{C_f l_f}{I_z} \end{bmatrix}$$

Vehicle Actuation

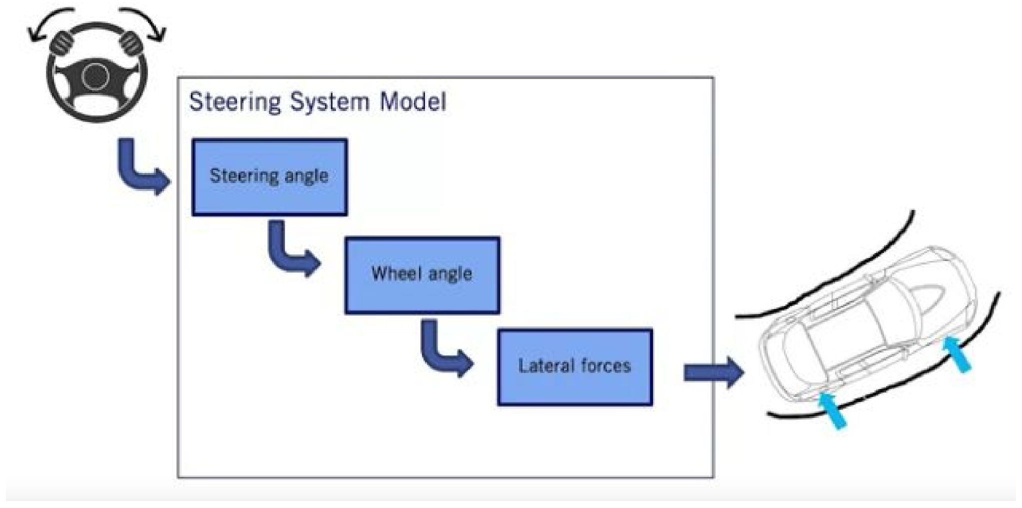
Coupled Lateral & Longitudinal



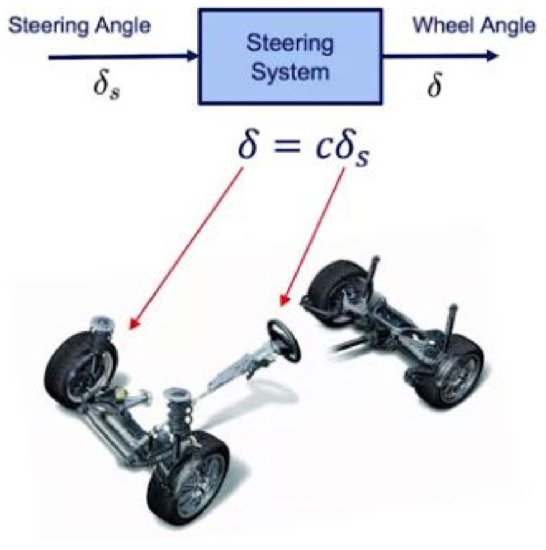
Main control task

To keep the vehicle on the defined path at the desired velocity.

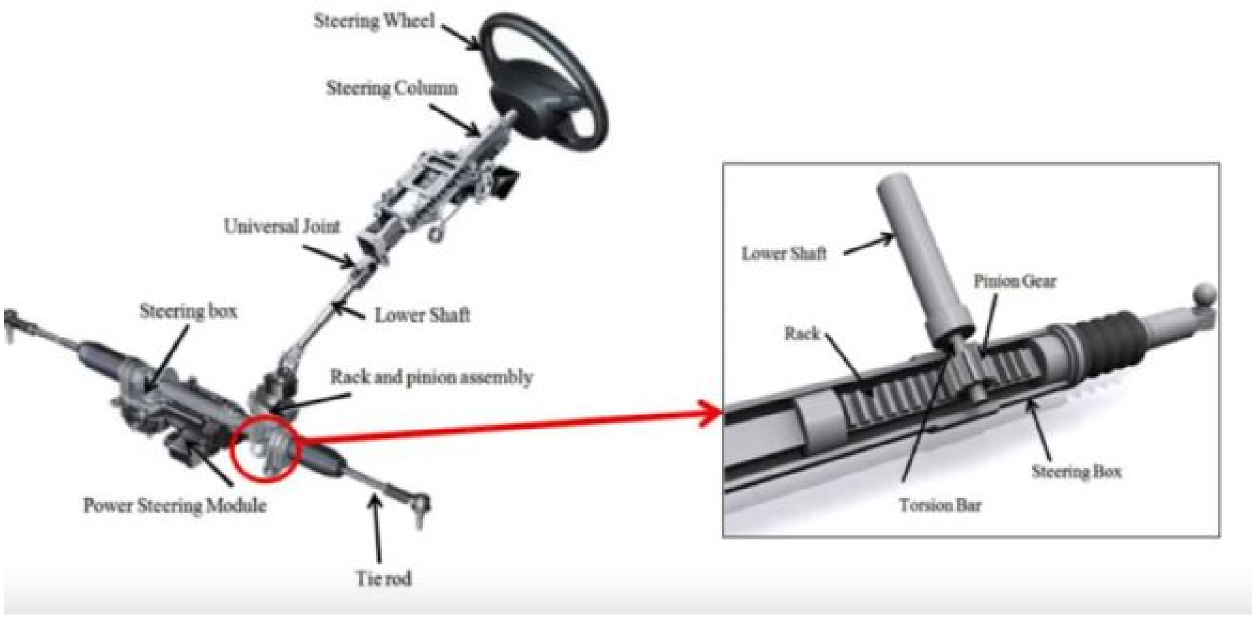
Steering model



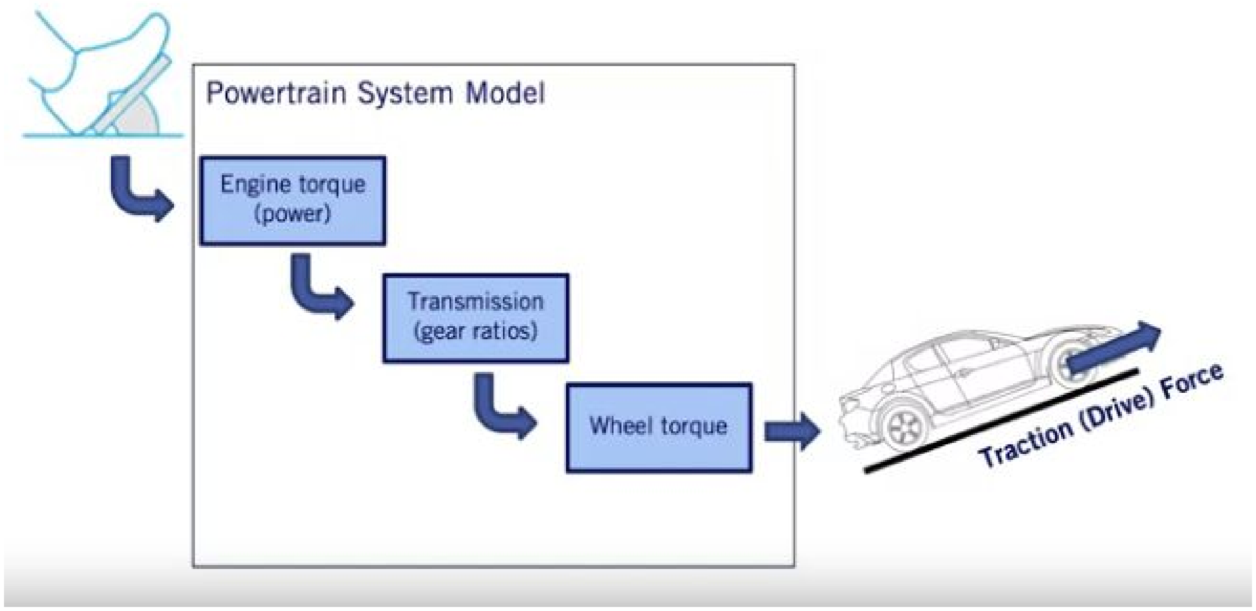
Simple steering model

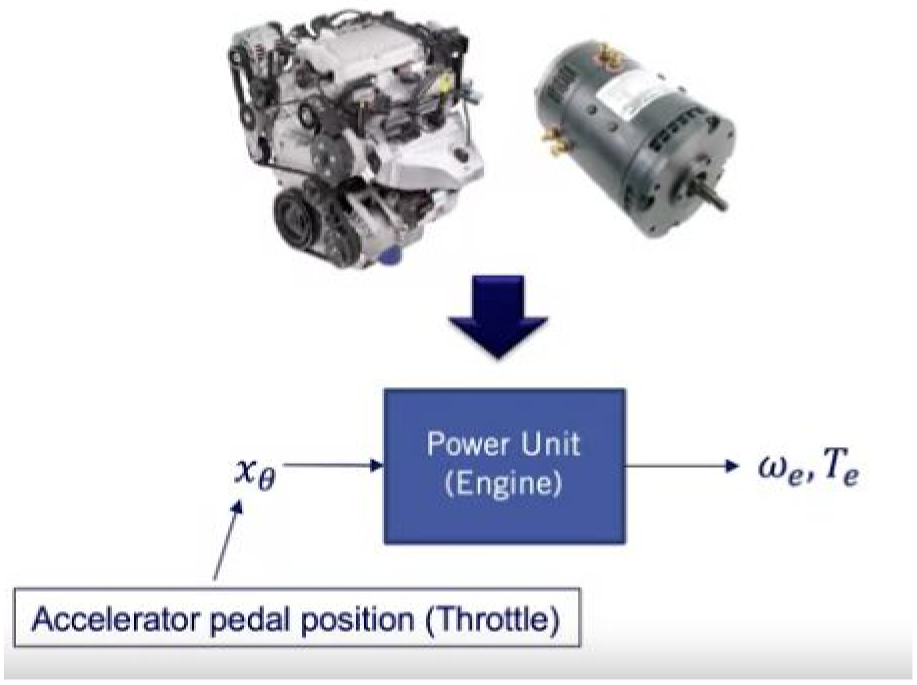


Real-world steering model

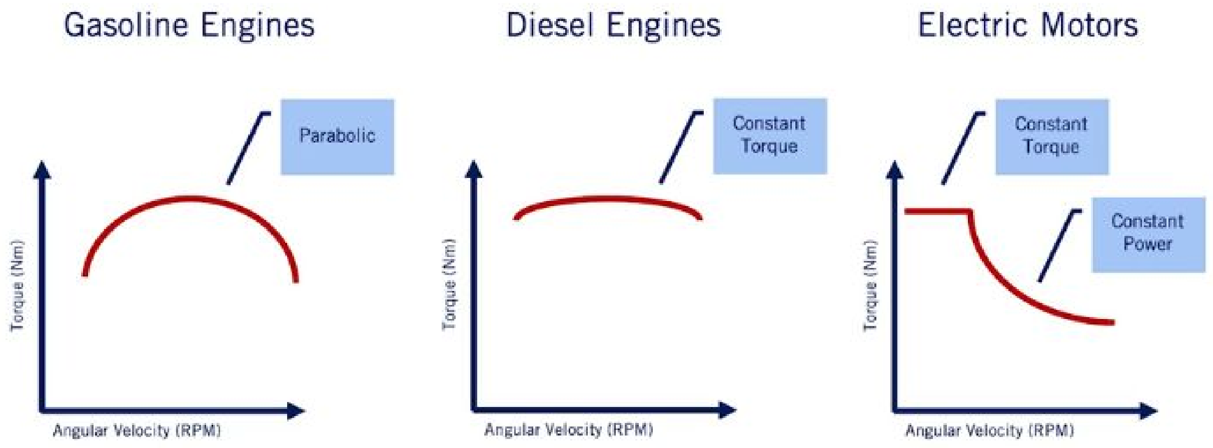


Throttling / accelerating

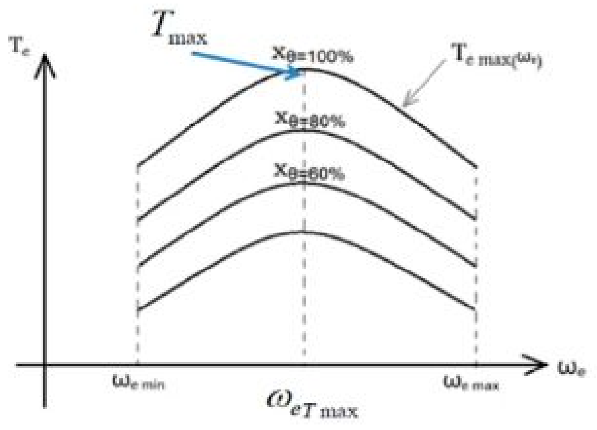




Characteristics plots



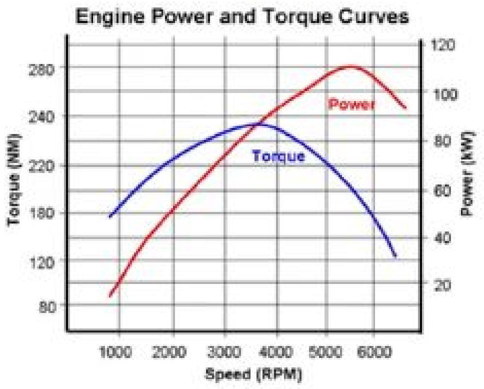
Typical Torque Curves for Gasoline Engines



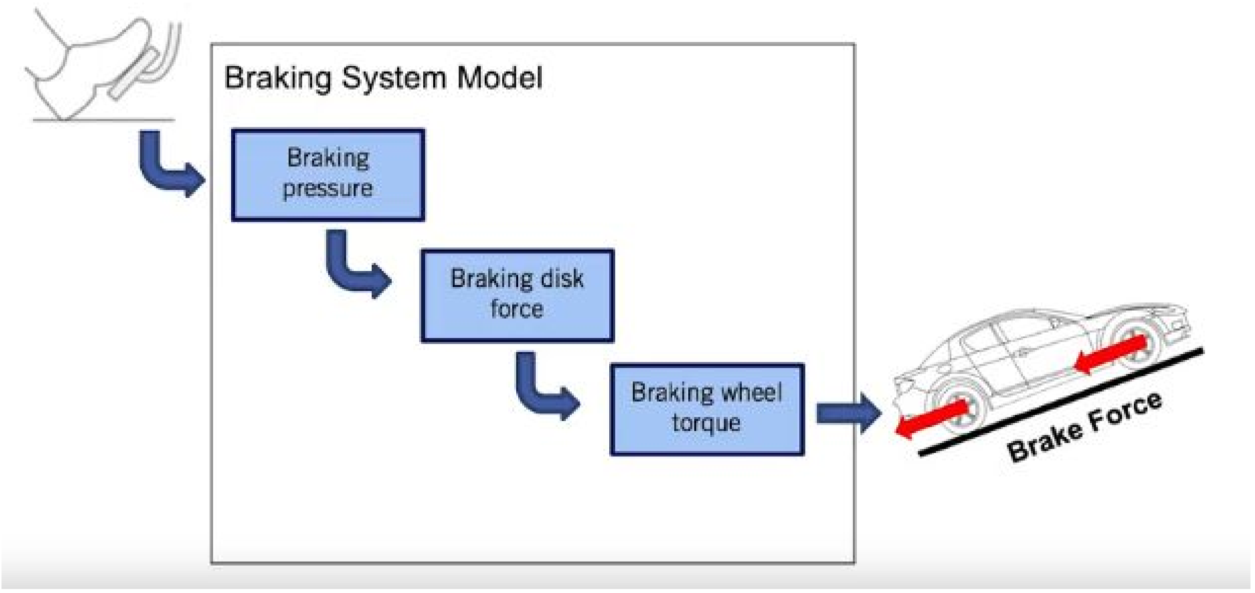
$$T_{e_{max}}(\omega_e) = A_0 + A_1\omega_e + A_2\omega_e^2$$

$$T_e(\omega_e, x_\theta) \approx x_\theta(A_0 + A_1\omega_e + A_2\omega_e^2)$$

Throttle position (percentage)



Decelerating



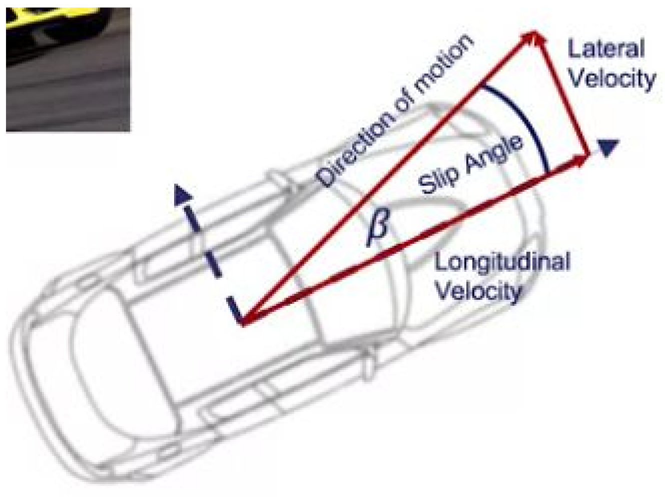
Basic functionality of braking system

- Shorten stopping distance

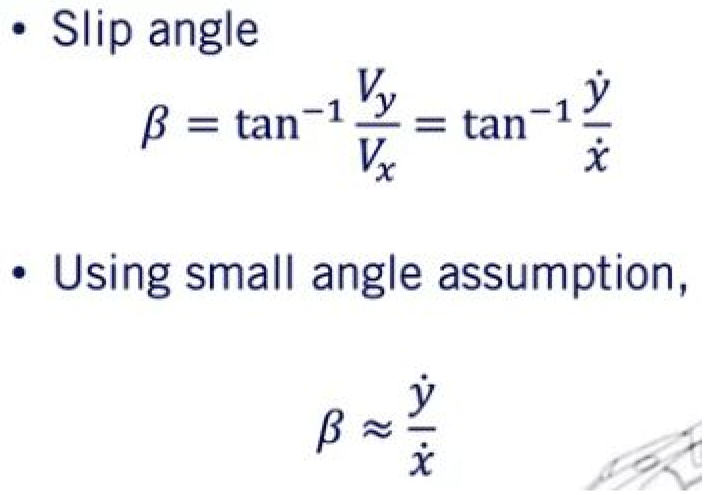
- Steering during braking through ABS system
- Stability during braking to avoid overturning

Tire modeling

The tire is the interface between the vehicle and road.
The wheel slip angle is the angle between the forward direction of the vehicle in the actual direction of its motion, which is denoted as Beta.



Tire slip angle is the angle between the direction in which a wheel is pointing and the direction in which it is actually travelling.



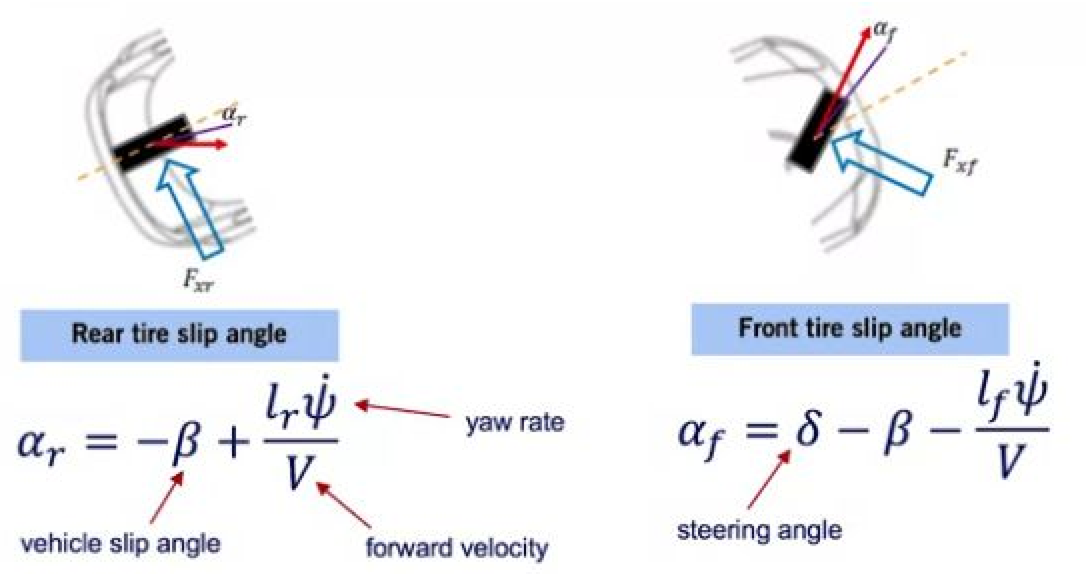
- Slip angle

$$\beta = \tan^{-1}\frac{V_y}{V_x} = \tan^{-1}\frac{\dot{y}}{\dot{x}}$$

- Using small angle assumption,

$$\beta \approx \frac{\dot{y}}{\dot{x}}$$

Relation between the tire slip angle and vehicle slip angle



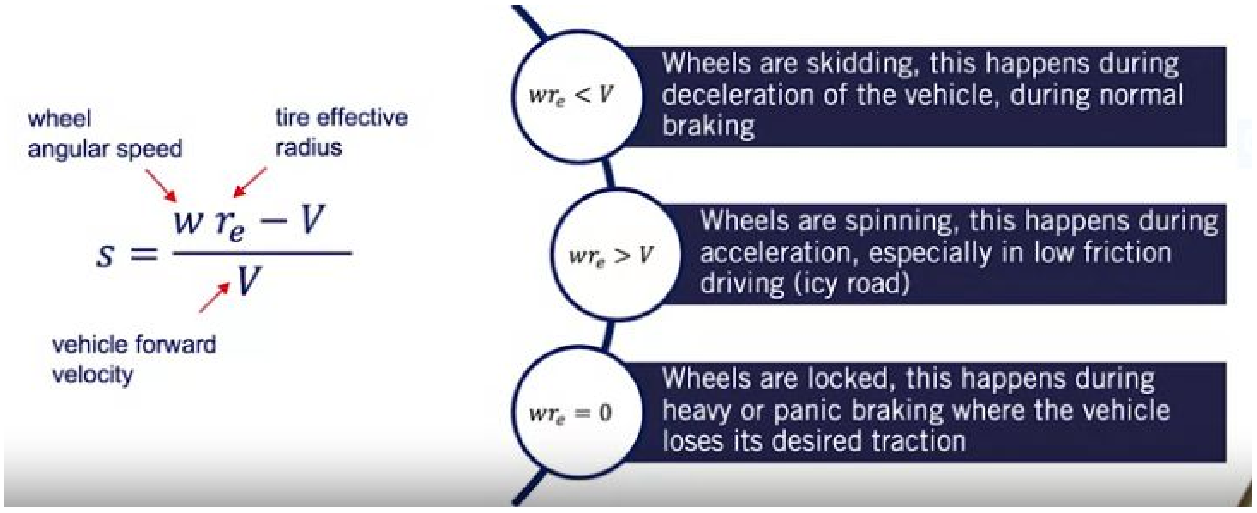
Rear tire slip angle

$$\alpha_r = -\beta + \frac{l_r \dot{\psi}}{V}$$

Front tire slip angle

$$\alpha_f = \delta - \beta - \frac{l_f \dot{\psi}}{V}$$

Slip ratio



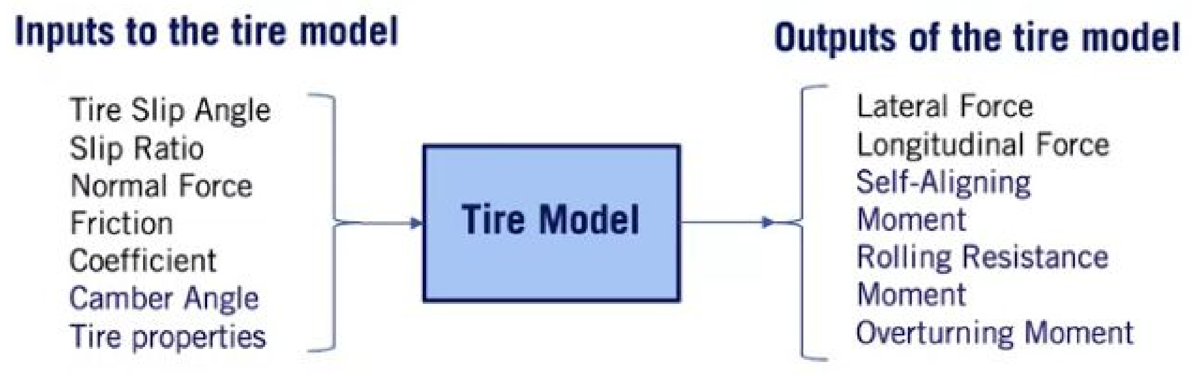
$$s = \frac{w\, r_e - V}{V}$$

$wr_e < V$ Wheels are skidding, this happens during deceleration of the vehicle, during normal braking

$wr_e > V$ Wheels are spinning, this happens during acceleration, especially in low friction driving (icy road)

$wr_e = 0$ Wheels are locked, this happens during heavy or panic braking where the vehicle loses its desired traction

Tire model



Inputs to the tire model

- Tire Slip Angle
- Slip Ratio
- Normal Force
- Friction Coefficient
- Camber Angle
- Tire properties

Outputs of the tire model

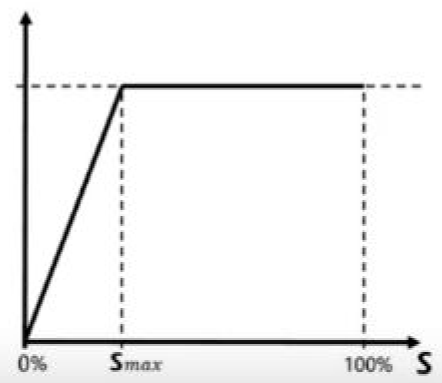
- Lateral Force
- Longitudinal Force
- Self-Aligning Moment
- Rolling Resistance Moment
- Overturning Moment

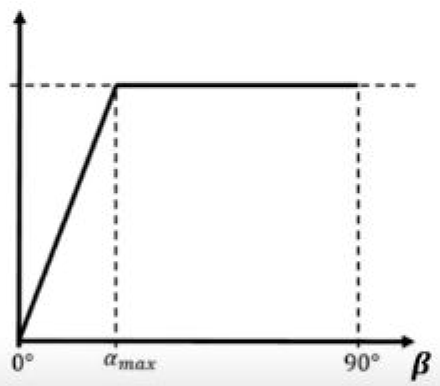
Tire modeling

- Analytical
    - Tire physical parameters are explicitly employed
    - Low precision, but simple
- Numerical
    - Look up tables instead of math equations
    - No explicit math form
    - Geometry and material property of tire are considered
- Parameterized
    - Need experiments for each specific tire
    - Formed by fitting model with experiment data
    - Match experimental data well
    - Used well for vehicle dynamics simulation studies and control design

Linear tire model

- Assumption: the relationship between slip angle and force is linear
    - Piecewise linear curves: $F(x) = \begin{cases} Cx & if\ |x| < x_{max} \\ F_{max} & if\ |x| \geq x_{max} \end{cases}$



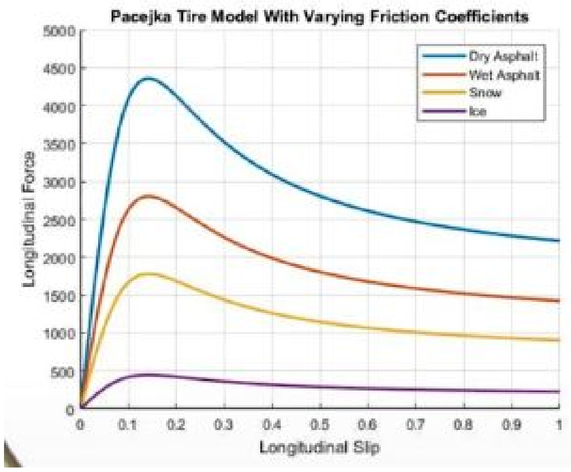


Pacejka Tire Model

- Also called Magic Formula tire model
  - Widely used in model-based control development.

tire vertical force

$$F(x, F_z) = D\sin(C\ \tan^{-1}(Bx - E(Bx - \tan^{-1}(Bx))))\mu F_z$$



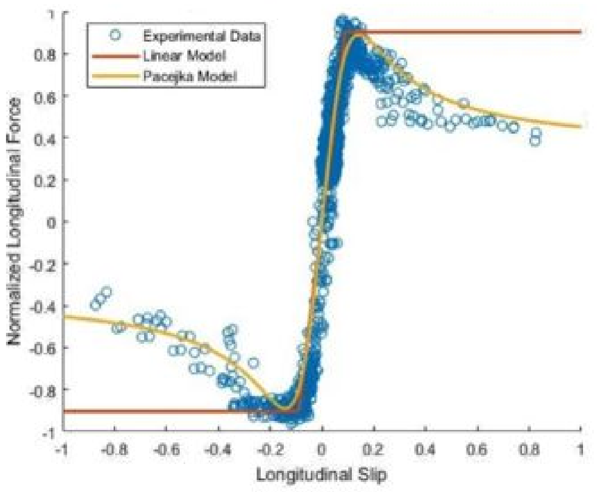
x could be either slip ratio or slip angle (in tire modeling)

road friction coefficient

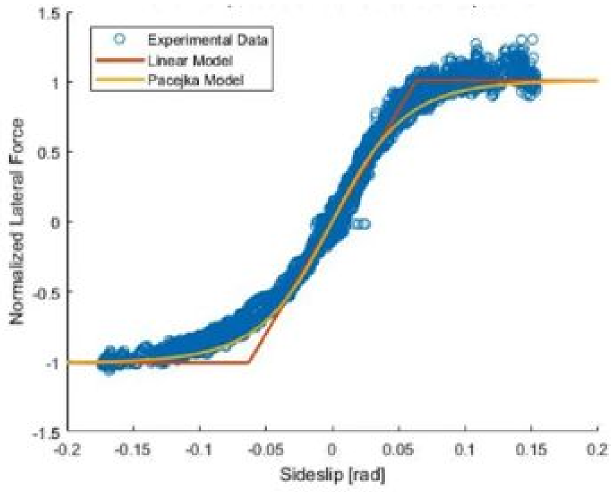
B – Stiffness Factor
C – Shape Factor
D – Peak Factor
E – Curvature Factor

Forces vs Slips

Normalized Longitudinal Force vs. Slip Ratio

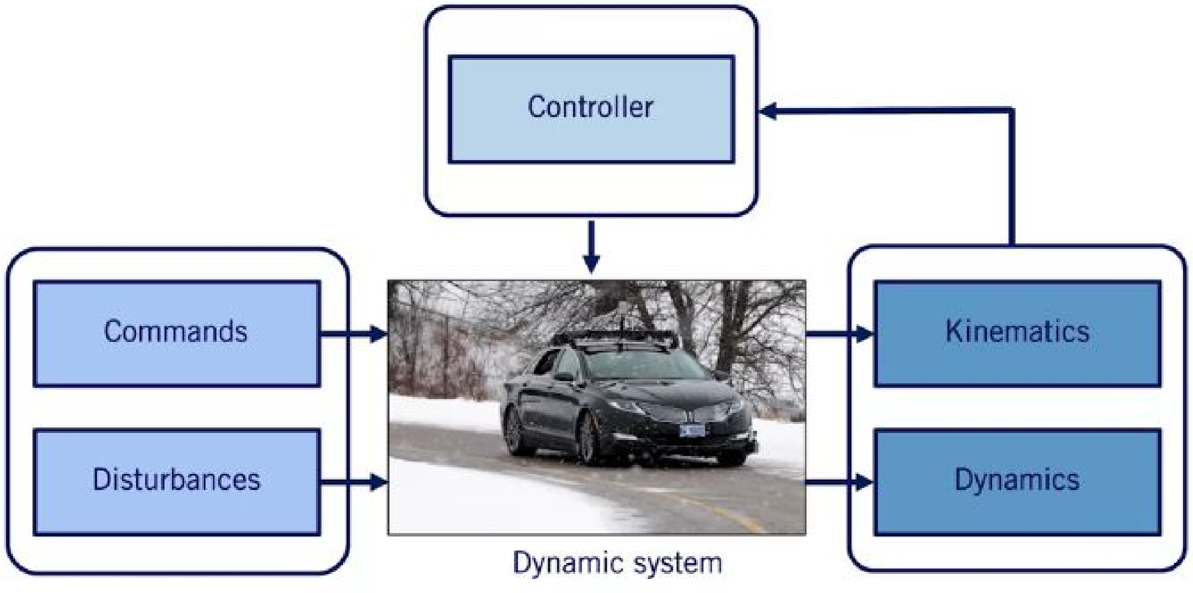


Normalized Lateral Force vs. Slip Angle



Proportional-Integral-Derivative Control (PID Control)

Control development



The dynamic and kinematic models for a vehicle aim to **capture how the dynamic system reacts** to input commands from the driver such as steering gas and break and how it reacts to disturbances such as wind, road surface and different vehicle loads. **The effects of the inputs and disturbances on the states** such as velocity and rotation rate of the vehicle **are defined by the kinematic and dynamic models** we developed.
The role of the controller then is to regulate some of these states of the vehicle by sensing the current state variables and then **generating actuator signals to satisfy the commands provided.**

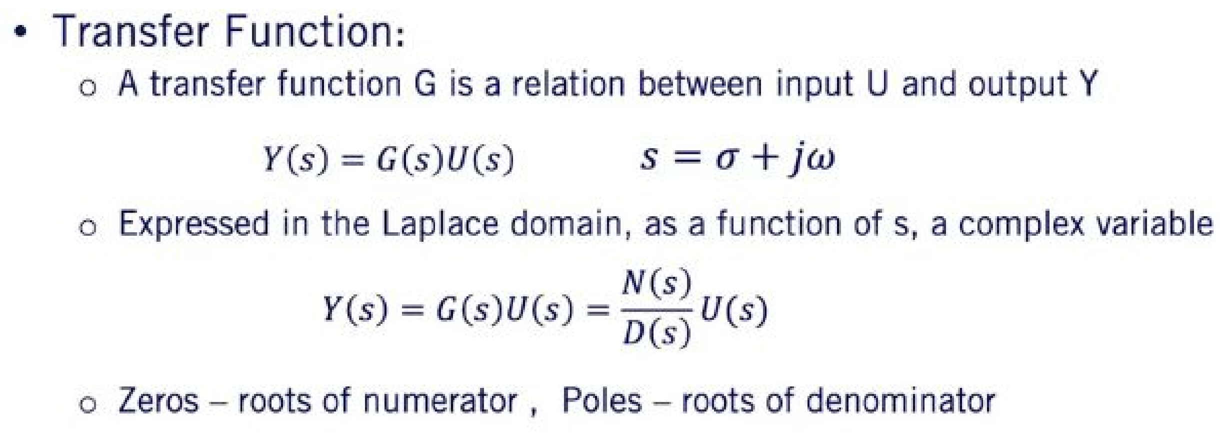
For longitudinal control, the controller sensing the vehicle speed and adjust the throttle and break commands to match the desired speed set by the autonomous motion planning system.

The plant or process model takes the actuator signals as the input and generates the output or state variables of the system. These outputs are measured by sensors and estimators are used to fuse measurements into accurate output estimates. The output estimates are compared to the desired or reference output variables and the difference or error is passed to the controller. **The controller can be seen as a mathematical algorithm that generates actuator signals** so that the error signal is minimized and the plant state variables approach the desired state variables.

Plant System or Process

System representation

- The plant system can be linear or nonlinear
- Plant representation: state-space form and transfer functions
    - State-space form which tracks the evolution of an internal state to connect the input to the output
    - Transfer function models the input and output directly
- Linear time-invariant systems can be expressed using transfer functions



- Transfer Function:
    - A transfer function G is a relation between input U and output Y

$$Y(s) = G(s)U(s) \qquad s = \sigma + j\omega$$

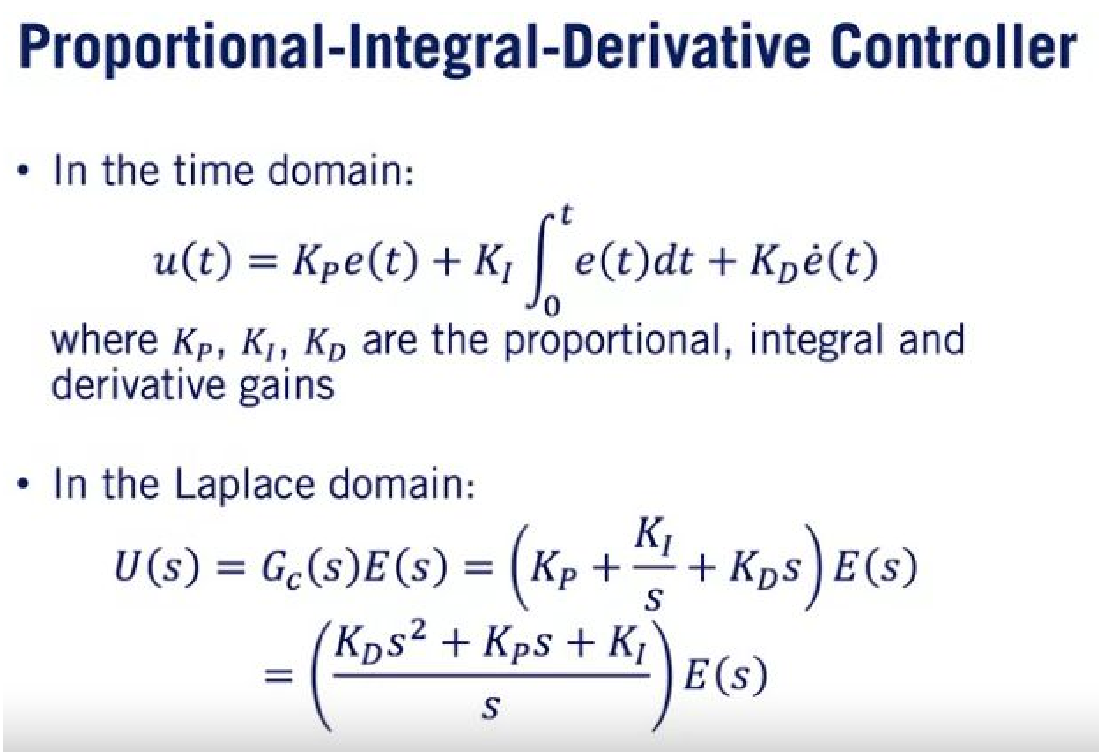
    - Expressed in the Laplace domain, as a function of s, a complex variable

$$Y(s) = G(s)U(s) = \frac{N(s)}{D(s)}U(s)$$

    - Zeros – roots of numerator , Poles – roots of denominator

Control or Compensator

- Control algorithms can vary from simple to complex
- Some simple algorithms often used in the industry:
    - Lead-lag controller
    - PID controller
- More complex algorithms
    - Nonlinear methods, feedback linearization, backstepping, sliding mode
    - Optimization methods, model prediction control



Proportional-Integral-Derivative Controller

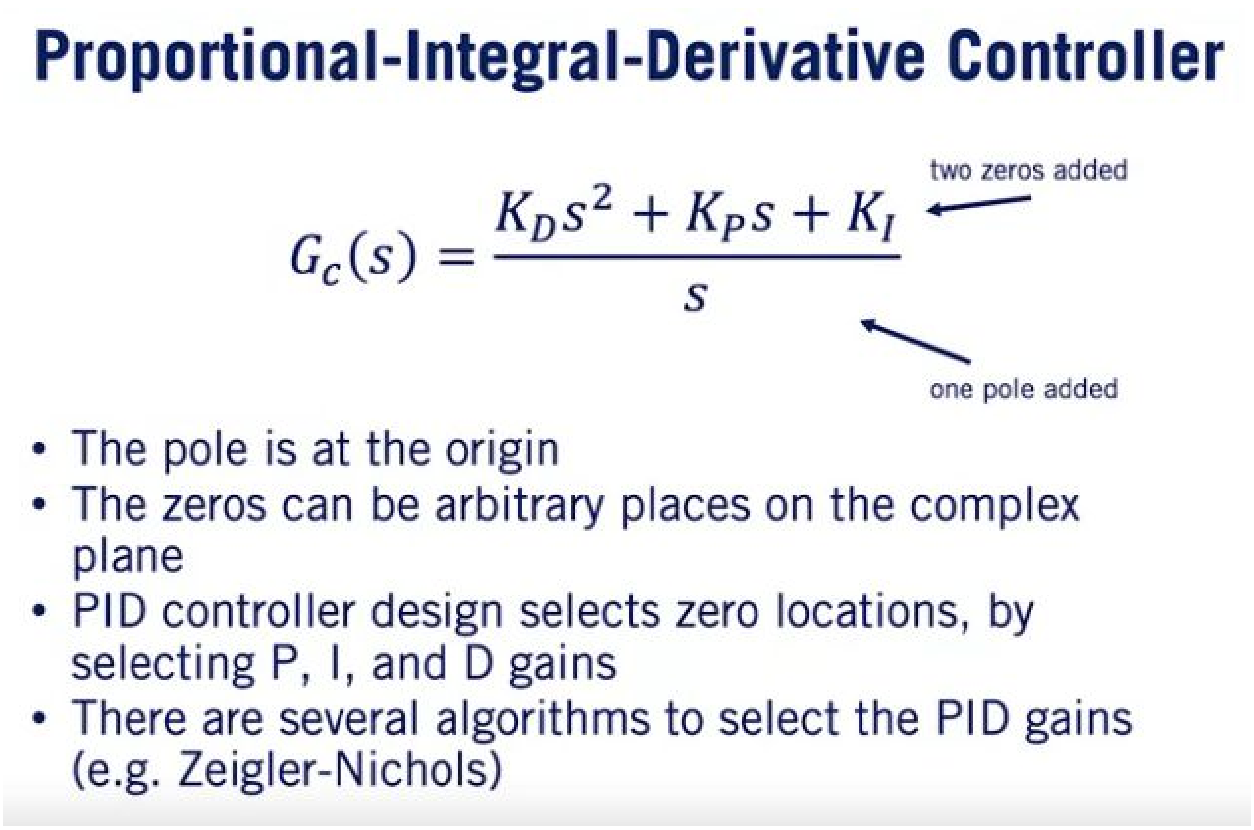
- In the time domain:

$$u(t) = K_P e(t) + K_I \int_0^t e(t)dt + K_D \dot{e}(t)$$

where K_P, K_I, K_D are the proportional, integral and derivative gains

- In the Laplace domain:

$$U(s) = G_c(s)E(s) = \left(K_P + \frac{K_I}{s} + K_D s\right)E(s)$$
$$= \left(\frac{K_D s^2 + K_P s + K_I}{s}\right)E(s)$$



Proportional-Integral-Derivative Controller

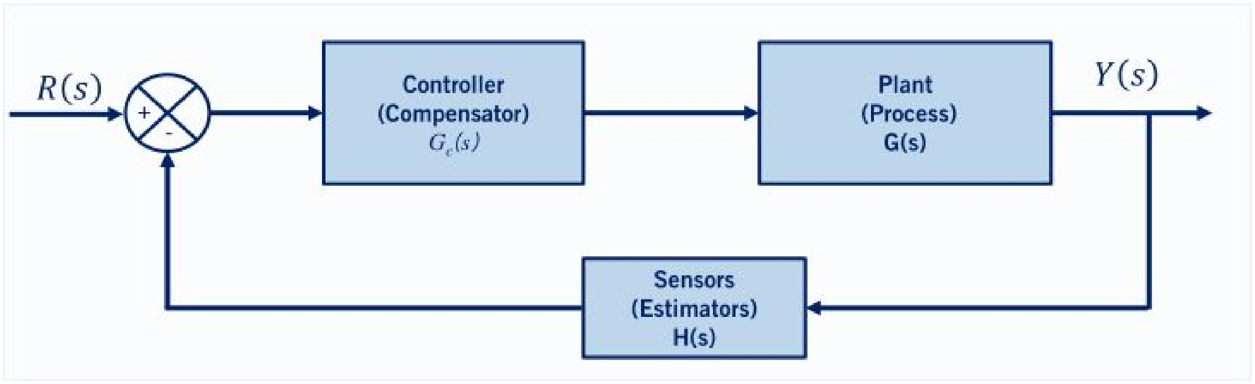
$$G_c(s) = \frac{K_D s^2 + K_P s + K_I}{s}$$

two zeros added

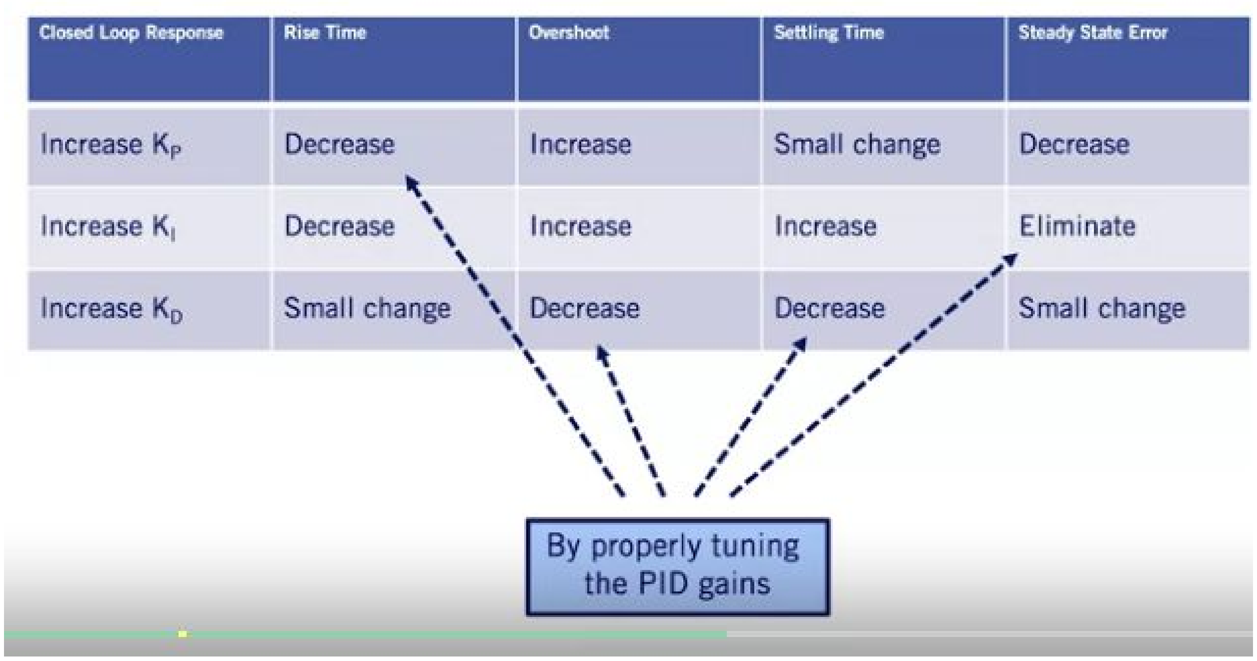
one pole added

- The pole is at the origin
- The zeros can be arbitrary places on the complex plane
- PID controller design selects zero locations, by selecting P, I, and D gains
- There are several algorithms to select the PID gains (e.g. Zeigler-Nichols)

Closed loop response denotes the response of a system when the controller decides the inputs to apply to the plant model. For a step input on the reference signal we can define the **rise time** as the time it takes to reach 90 percent of the reference value. The **overshoot** as the maximum percentage the output exceeds this reference. The **settling time** as the time to settle to within five percent of the reference and the steady-state error as the error between the output and the reference at steady-state.



Characteristics of P, I, D gains

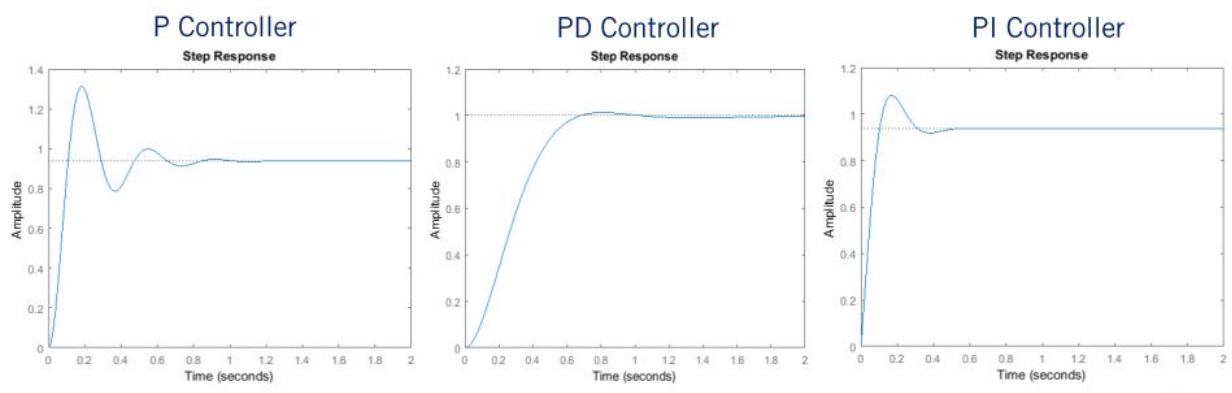


Closed Loop Response	Rise Time	Overshoot	Settling Time	Steady State Error
Increase K_P	Decrease	Increase	Small change	Decrease
Increase K_I	Decrease	Increase	Increase	Eliminate
Increase K_D	Small change	Decrease	Decrease	Small change

By properly tuning the PID gains

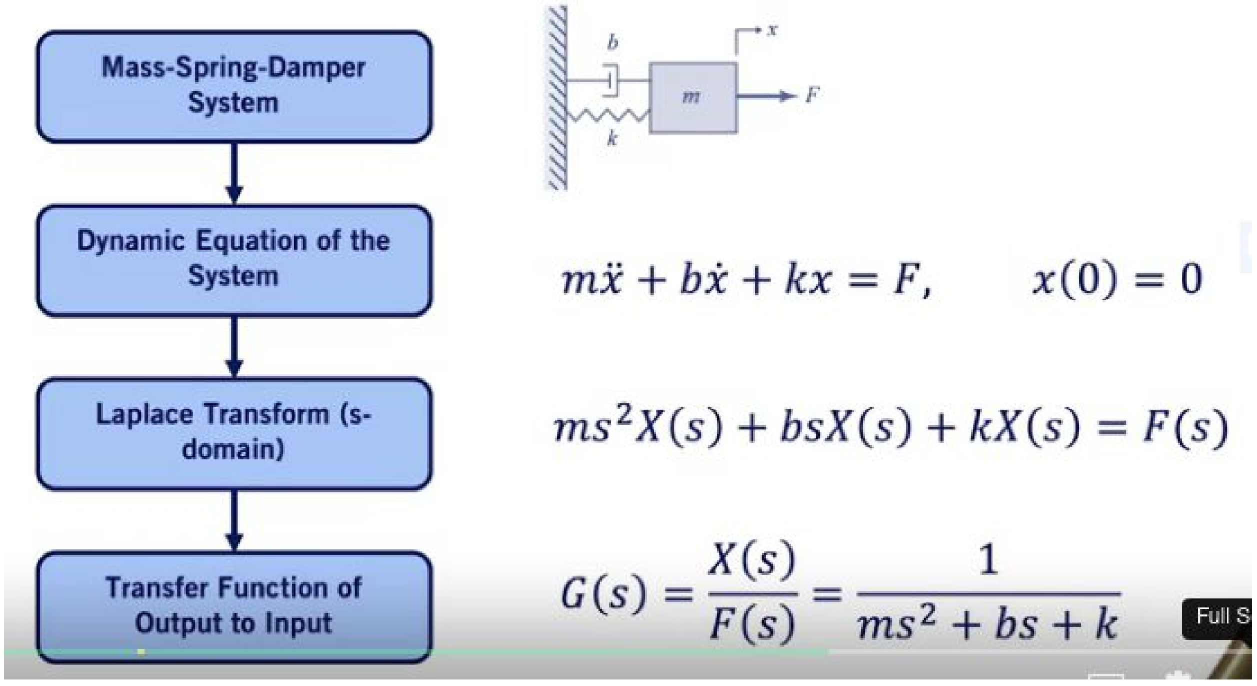
By carefully tuning the controller gains, we can use the benefits of all three to eliminate overshoot and still maintain very short rise and settling times.

Step response



$$G_P(s) = K_P \qquad G_{PD}(s) = K_P + sK_D \qquad G_{PI}(s) = K_P + \frac{K_I}{s}$$

Example: Second order system



$$\begin{tikzcd}[row sep=large]
\text{Mass-Spring-Damper System} \arrow[d] \\
\text{Dynamic Equation of the System} \arrow[d] \\
\text{Laplace Transform (s-domain)} \arrow[d] \\
\text{Transfer Function of Output to Input}
\end{tikzcd}$$

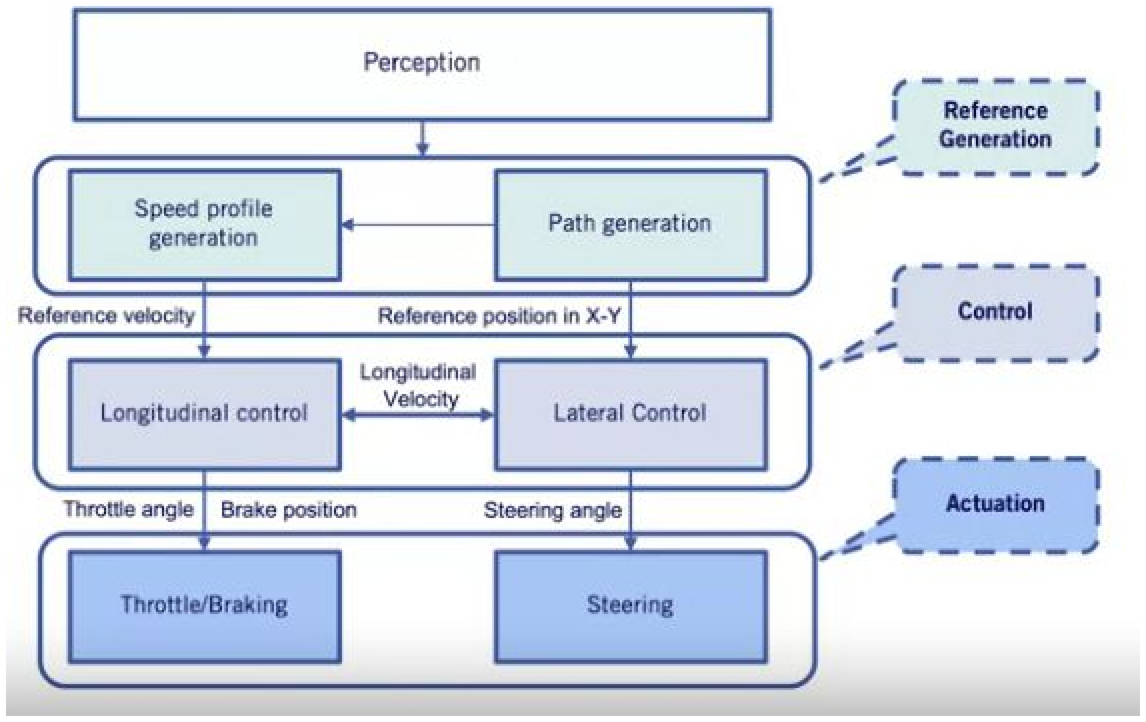
$$m\ddot{x} + b\dot{x} + kx = F, \qquad x(0) = 0$$

$$ms^2X(s) + bsX(s) + kX(s) = F(s)$$

$$G(s) = \frac{X(s)}{F(s)} = \frac{1}{ms^2 + bs + k}$$

Classical control: Textbook by Prof. Bruce Francis (University of Toronto), covers Laplace Transforms, Bode Diagrams, Nyquist Plots

Longitudinal Speed Control with PID

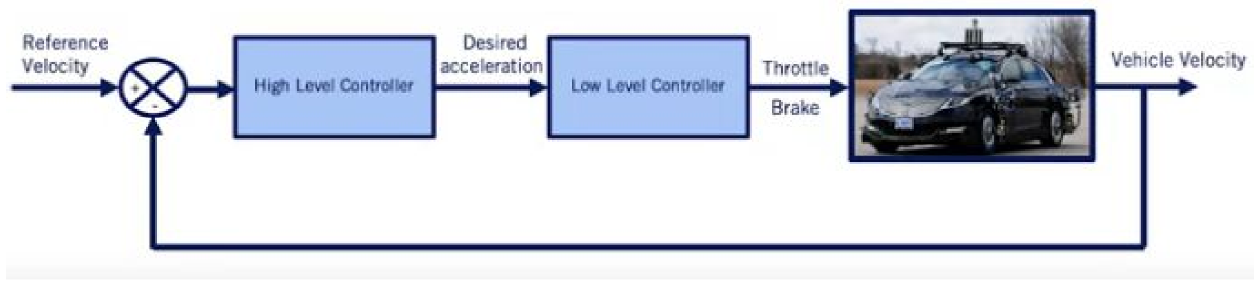


.

- The first section is the perception of the road and the environment. This perception is captured by sensors and generates the input references for our system.
- In the second layer, we have both path generation and speed profile generation, which in automotive circles is referred to as the drive cycle. These profiles are generated through the motion planning process. The path and the speed profiles are the reference inputs needed by our controllers. For longitudinal control, define the set point's, acceleration and deceleration that we'd like to be able to track precisely.
- For both the lateral and longitudinal control of an autonomous vehicle, the only task that needs to be performed is to follow the plan as precisely as possible, and thereby minimize the error between the actual and reference path and speed. All other tasks required for autonomous driving or done by other parts of the system.
- Finally, the controllers generate the input commands or actuator signals for the vehicle. As we've seen in the previous module, these include the steering for the lateral control and the throttle and break commands for longitudinal control.

Examples: Cruise control

Speed of the vehicle is controlled (by throttling and braking) to be kept at the reference speed.



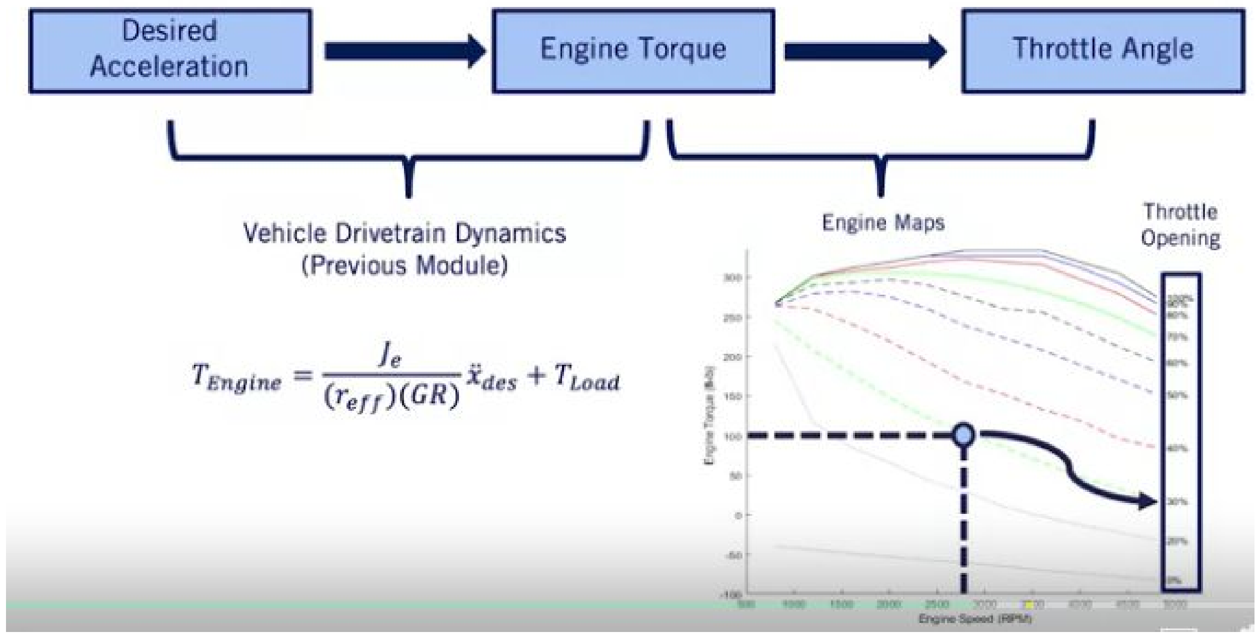
The controller can be split into two levels;
A high level and a low level controller, although the low level controller is not essential to the control task. The high level controller takes the difference between
the set point velocity and the vehicle actual velocity, and generates the desired vehicle acceleration to close the gap. The low-level controller gets the vehicle acceleration and **generates a throttle or breaking actuation** to track the reference acceleration. In practice, this two-stage approach allows us to go beyond just PID control and impose limits or profiles directly on the accelerations that are requested of the vehicle in order to maintain speed. It also allows us to separate the use of engine maps for generating a desired torque given the engine state from the cruise control input response.
Upper level controller: Determines the desired acceleration for the vehicle (based on the reference and actual velocity).

Desired Acceleration   Reference Velocity   Vehicle Velocity

$$\ddot{x}_{des} = K_P(\dot{x}_{ref} - \dot{x}) + K_I \int_0^t (\dot{x}_{ref} - \dot{x})dt + K_D \frac{d(\dot{x}_{ref} - \dot{x})}{dt}$$

PID Gains

Lower level controller: throttle input is calculated such that the vehicle track the desired acceleration determined by the upper level controller
Assumptions:
- Only throttle actuations is considered, no braking
- The torque converter is locked (gear 3+)
- The tire slip is small (gentle longitudinal maneuvers)



The low-level controllers seeks to **generate the desired acceleration** from the high level controller by increasing or decreasing the torque produced by the engine. This is controlled by the throttle angle, but is governed by the power train dynamics and the engine map, making it a nonlinear problem that can be a challenge for classic control methods. Instead, the desired acceleration is translated to a torque demand, and the torque demand is then converted to a throttle angle command.

We can **rearrange vehicle drivetrain dynamics equation** to solve for the desired engine torque, given known load torques and the desired acceleration of the vehicle. Then, **the steady-state engine map**, which is generated in testing the engine at different operating points can be used to **determine the throttle angle** needed to produce the amount of torque demand required. In these standard maps, the desired engine torque and the current engine RPM (revolutions per minute) **define the required throttle position**, and can be interpolated if needed. This approach is a data-driven approximation, but it works quite well in practice. The approximation comes from the fact that the data points in the map are steady-state points while the power train is continuously changing its operating point to meet the current driving conditions.

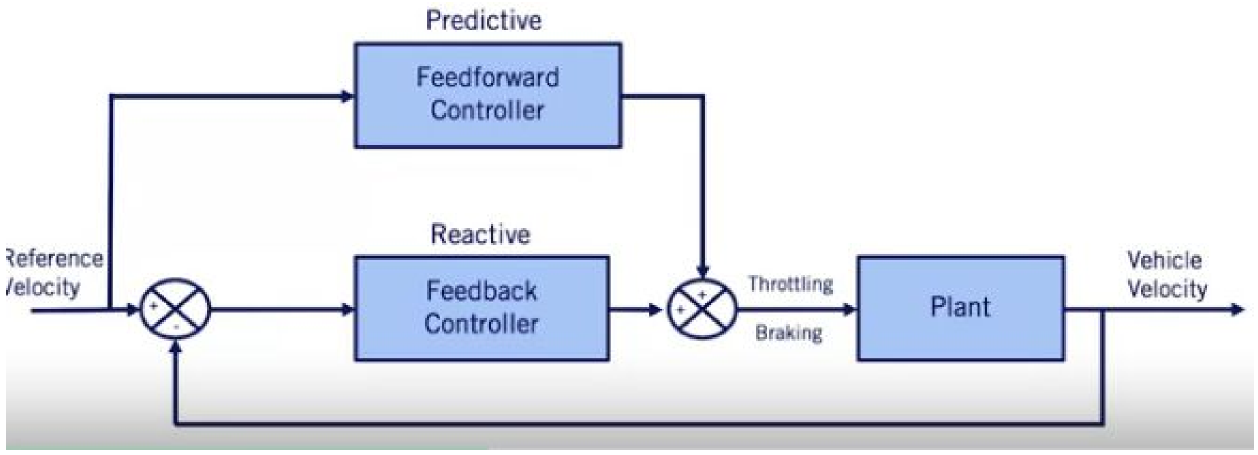
Finally, we can put the pieces of our vehicle controller together and simulate the control response to a step change in desired speed of our dynamic vehicle models with PID controllers. The PID gains are tuned by trial and error so that the vehicle speeds follow the reference velocity of 30 meters per second or 108 kilometers per hour.

Because of the **engine map non-linearity**, we see some interesting artifacts in the vehicle response as it closes in on the reference speed. Gear changes could also cause big challenges for pure PID control.

Feedforward speed control

Feedback: closed loop
Feedforward: open loop
Combined feedforward and feedback control to improve performance

- Feedforward controller provides predictive response, non-zero offset
- Feedback controller corrects the response, compensate for disturbance and errors in the model



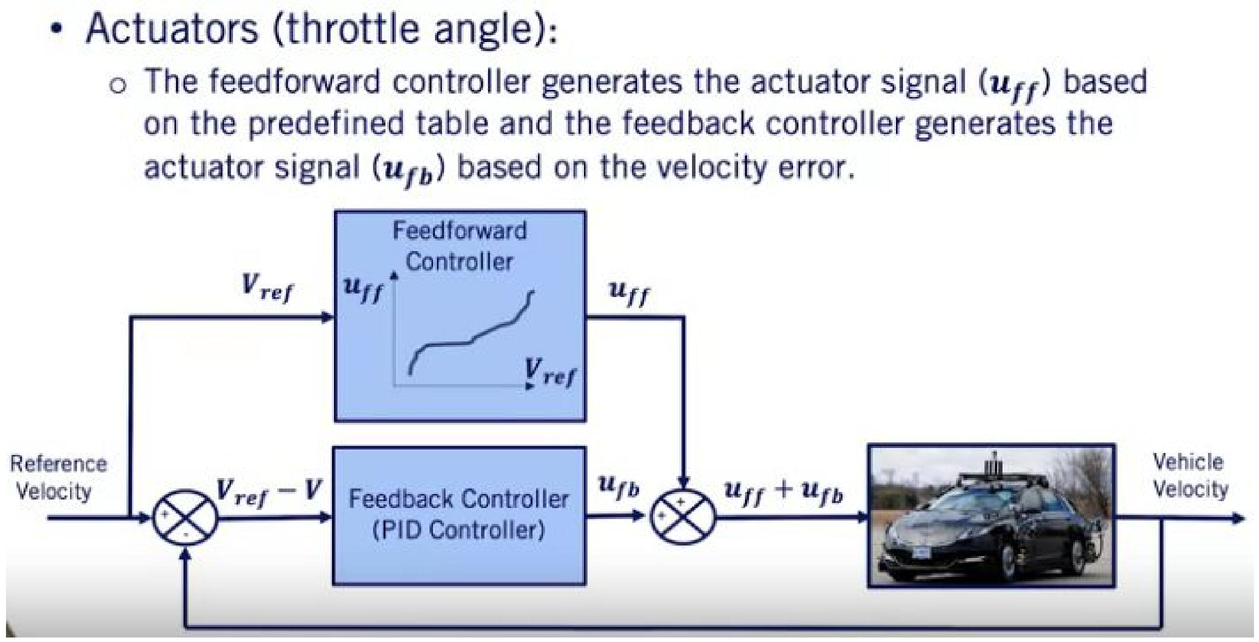
The combination of feedback and feedforward control is widely used because of this complementary relationship. Because autonomous vehicles require non-zero steering commands to maintain a constant radius turn and a constant throttle or brake command to maintain constant speed or deceleration rates, feedforward commands are extremely beneficial in improving tracking performance in automated driving.

Throttling and braking:

- The output of the feedforward and feedback control blocks are the throttling or braking signals to accelerate or decelerate the vehicle (plant) to keep the vehicle velocity close to the reference velocity.

The reference speed or drive cycle is defined by a higher level planner. And it is desirable that the vehicle follows the reference velocity precisely. The reference velocity is the input to the feedforward block, and the velocity error is the input to the feedback or PID control block. Both controllers produce two vehicle actuation signals, the throttle and the brake commands. Note that there is no low-level controller included in this block diagram, as we had in the pure PID feedback control from the previous video. The role of the low-level controller achieving the desired acceleration through the use of a mapping from accelerations to engine commands is now going to be handled by the feedforward block.

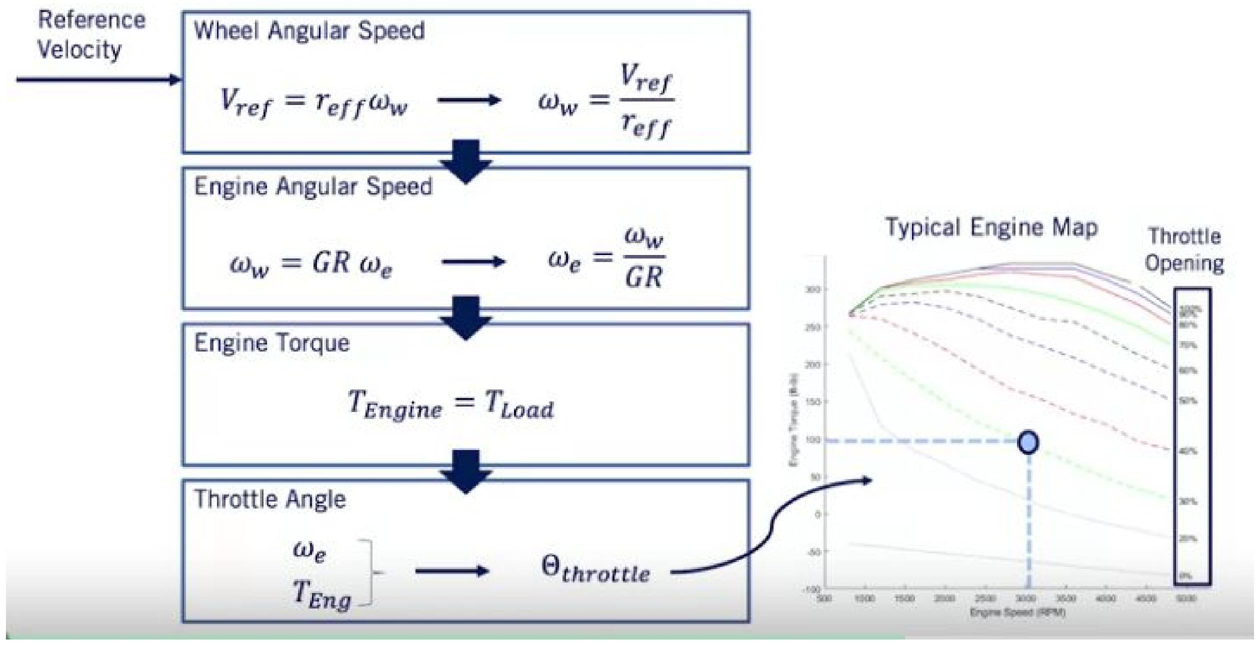
Controller actuator



- Actuators (throttle angle):
  - The feedforward controller generates the actuator signal ($\boldsymbol{u_{ff}}$) based on the predefined table and the feedback controller generates the actuator signal ($\boldsymbol{u_{fb}}$) based on the velocity error.

The feedforward block gets only the reference signal as input, and its primary objective is to accurately set the inputs of the plan. To do this we can convert the entire longitudinal dynamics model into a fixed lookup table or reference map, that maps the reference velocity to the corresponding actuators signals assuming the vehicle is at steady state. This feedforward approach works well at steady state, but ignores the internal dynamics of the vehicle powertrain, and must also rely on the current vehicle state estimate to resolve some of the forces and dynamic models used.

Feedforward table


Reference Velocity
Wheel Angular Speed
$V_{ref} = r_{eff}\omega_w \longrightarrow \omega_w = \frac{V_{ref}}{r_{eff}}$
Engine Angular Speed
$\omega_w = GR\,\omega_e \longrightarrow \omega_e = \frac{\omega_w}{GR}$
Engine Torque
$T_{Engine} = T_{Load}$
Throttle Angle
$\omega_e, T_{Eng} \longrightarrow \Theta_{throttle}$
Typical Engine Map
Throttle Opening
Engine Torque (lb-ft)
Engine Speed (RPM)

Comparing PID and PID+feedforward

The key difference between the two responses is visible as the reference speed changes. Because the PID controller needs errors to exist before it can correct them, its response lags the feedforward approach, which immediately applies the relevant input reference values. The feedforward tracking is still not perfect, however, as the vehicle response is ultimately governed by its inertia, and the feedforward approach we've presented relies on steady state modeling of the car. As the feedforward model becomes more precise, the feedback components can focus purely on disturbance rejection, and speed profile tracking can be done with consistent precision.

Lateral vehicle control

One of the main concerns in autonomous vehicles is ensuring the vehicle can precisely follow a predefined path, executing the motion plan devised in the higher level planning module.

Lateral control for an automobile

- Define error relative to desired path from vehicle position
- Select a control design between the vehicle position and the appropriate desired path coordinates that drives errors to zero and satisfies steering angle limits and dynamics limitations of vehicle and desired ride characteristics such as maximum lateral acceleration and minimum jerk.

- Add dynamic considerations as control command must be cognizant of the available tire forces and not exceed the capabilities of the vehicle when correcting for tracking errors

The reference path

The reference path is a fundamental interface between the planning system in the lateral controller

- Track
    - Straight line segments

The easiest approach is to define a sequence of straight line segments by requiring a sequence of end point vertices that are connected linearly. This path definition can be very compact and easy to construct,assuming points are well spaced and the environment allows for mostly straight line motion, as in a Manhattan grid of roadways. However, the path includes heading discontinuities, which make precise tracking a challenge with a steered vehicle.

-
    - Waypoints

A refinement of the line segment approach is to provide a series of tightly spaced waypoints. This spacing is usually fixed in terms of distance or travel time. The relative position of the waypoints can be restricted to satisfy an approximate curvature constraint. Waypoint paths are very common, as they are easy to work with and can be directly constructed from state estimates or GPS waypoints collected in earlier runs of a particular route.

-
    - Parameterized curves

It is also possible to define a path using a sequence of continuous parameterized curves, which can be either drawn from a fixed set of motion primitives or can be identified through optimization during planning. These curves provide the benefit of **continuously varying motion**, and can be constructed to have smooth derivatives to aid in the consistency of error and error rate calculations.

- Main goal
    - Heading path alignment
    - Elimination of offset to path

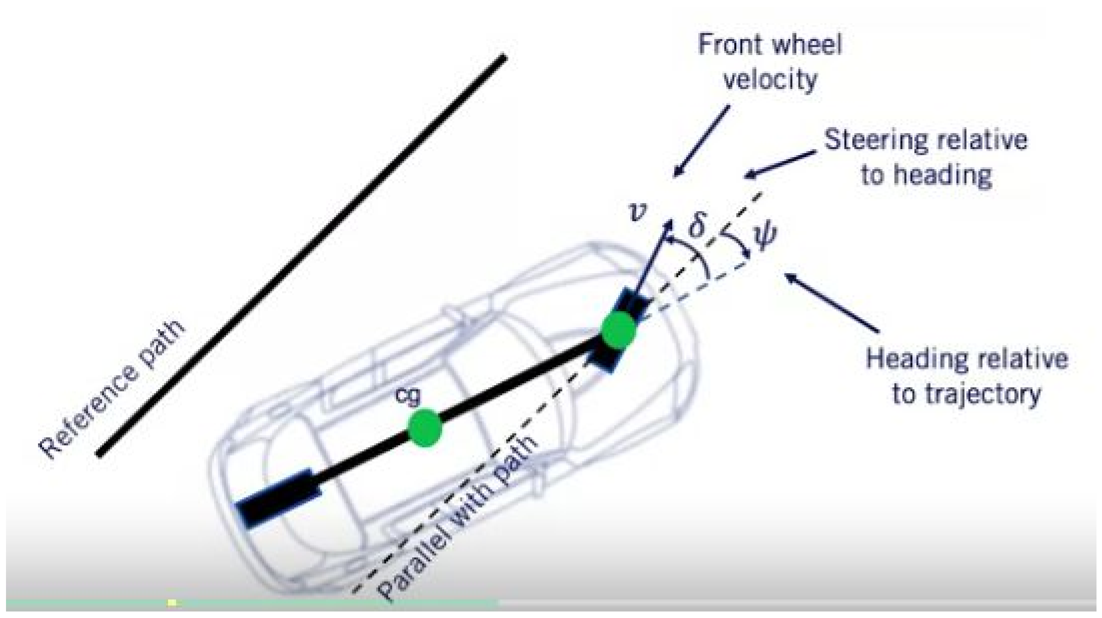
Two types of control design

- Geometric controllers: rely on the geometry and coordinates of the desired path and the kinematic models of the vehicle.
    - Pure pursuit (carrot following)
    - Stanley
- Dynamic controllers
    - Model Predictive Controller (MPC)
        - Able to handle a wide variety of constraints
        - identify optimized solutions that consider more than just the current errors.
    - Other control systems
        - Sliding mode, feedback linearization

Plant model

Vehicle (bicycle) model & parameters

- All state variables and inputs defined relative to the center of front axle

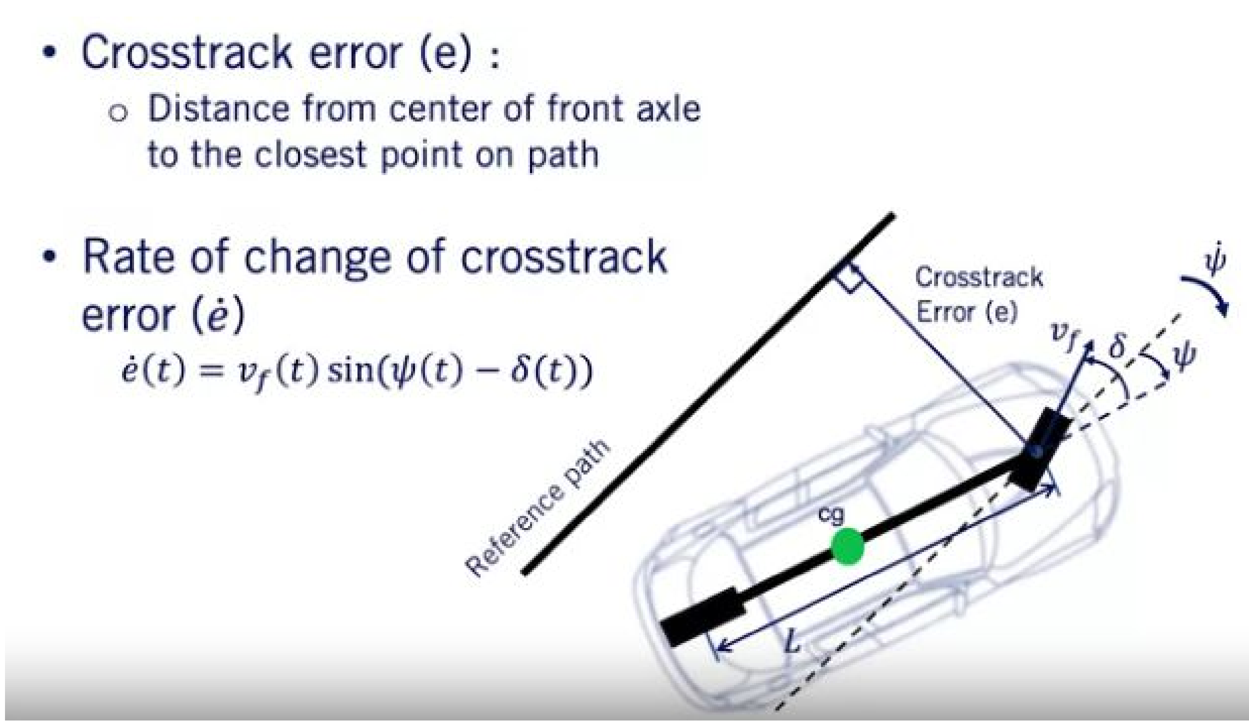


Driving controller

- Controller error terms
    - Heading error
        - Component of velocity perpendicular to trajectory divided by ICR radius
        - Desired heading is zero

$$\cancel{\dot{\psi}_{des}}(t) - \dot{\psi}(t) = \frac{v_f(t)\sin\delta(t)}{L} \quad \Rightarrow \quad \dot{\psi}(t) = \frac{-v_f(t)\sin\delta(t)}{L}$$

    - Crosstrack error

- 

- Crosstrack error (e) :
  - Distance from center of front axle to the closest point on path
- Rate of change of crosstrack error ($\dot{e}$)

$$\dot{e}(t) = v_f(t)\sin(\psi(t) - \delta(t))$$

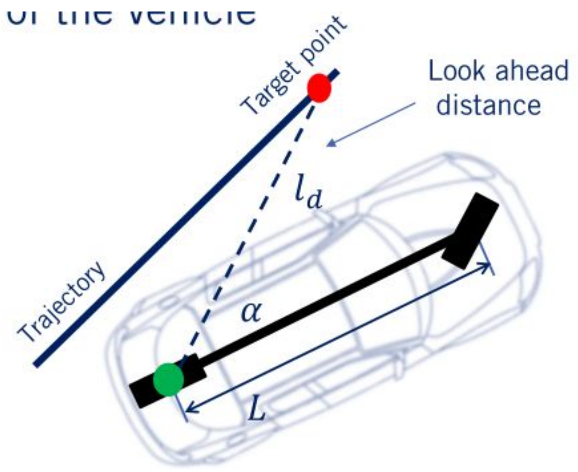
The rate of change of the crosstrack error can be calculated by extracting the lateral component of the forward velocity. From this equation, we can see that as the velocity increases, the crosstrack error changes more quickly, meaning that smaller steering angles are needed to correct for the same size crosstrack errors.

Reference

J. Jiang and A. Astolfi, "Lateral Control of an Autonomous Vehicle," in IEEE Transactions on Intelligent Vehicles, vol. 3, no. 2, pp. 228-237, June 2018. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8286943&isnumber=8363076

Geometric steering control - pure pursuit
- It exploits geometric relationship between the vehicle and the path resulting in compact control law solutions to the path tracking problem
- Use of reference point on path to measure error of the vehicle, can be ahead of the vehicle

Pure pursuit
- It consists of geometrically calculating the trajectory curvature
- Connect the center of rear axle location to a target point on the path ahead of the vehicle



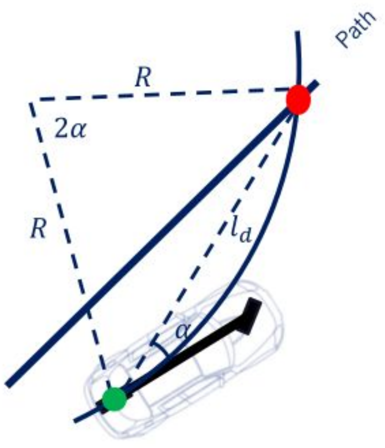
- Steering angle determined by target point location and angle between the vehicle's heading direction and lookahead direction

$$\frac{l_d}{\sin 2\alpha} = \frac{R}{\sin\left(\frac{\pi}{2} - \alpha\right)}$$

$$\frac{l_d}{2\sin\alpha\cos\alpha} = \frac{R}{\cos(\alpha)}$$

$$\frac{l_d}{\sin\alpha} = 2R$$

$$\kappa = \frac{1}{R} = \frac{2\sin\alpha}{l_d} \quad \text{Path curvature}$$



-
- Using the bicycle model the steering angle is calculated as

$$\underbrace{\kappa = \frac{2\sin\alpha}{l_d} \qquad \delta = \tan^{-1}\kappa L}$$

$$\delta = \tan^{-1}\left(\frac{2L\sin\alpha}{l_d}\right)$$

-
- Crossback error (e) is defined as the lateral distance between the heading vector and the target point

$$\left.\begin{array}{l} \sin\alpha = \dfrac{e}{l_d} \\[2ex] \kappa = \dfrac{2\sin\alpha}{l_d} \end{array}\right\} \quad \kappa = \frac{2}{l_d^2}e$$

- Pure pursuit is a proportional controller of the steering angle operating on a crosstrack error some look ahead distance in front of the vehicle
- The proportional gain $^2/_{l_d^2}$ can be tuned at different speeds (the l_d being assigned as a function of vehicle speed)
- • Lookahead l_d is assigned as a linear function of vehicle speed: $l_d = K_{dd}v_f$

$$\underbrace{\delta = \tan^{-1}\left(\frac{2L\sin\alpha}{l_d}\right) \qquad \kappa = \frac{2}{l_d^2}e}$$

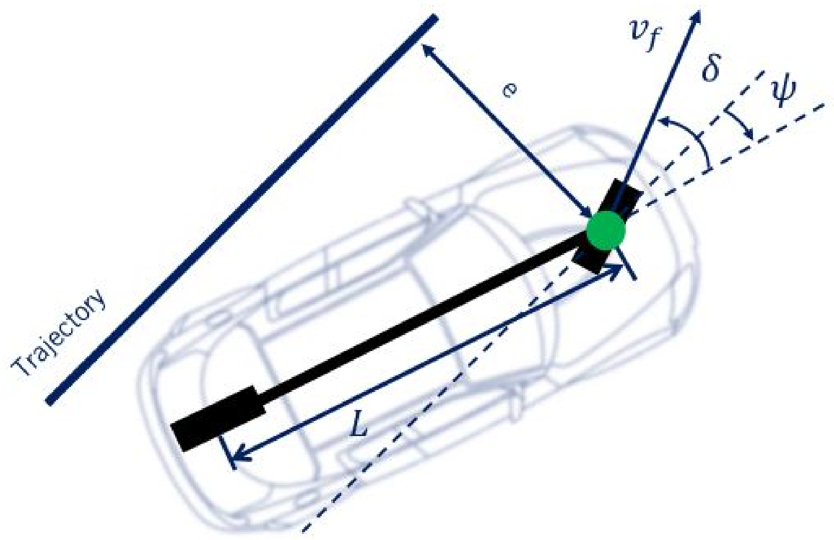
$$\delta = \tan^{-1}\left(\frac{2L\sin\alpha}{K_{dd}\underset{\substack{\nwarrow \\ \text{Forward velocity}}}{v_f}}\right)$$

Stanley controller method

- Uses the center of the front axle as a reference point
- Look at both the error in heading and the error in position related to the closest point on the path
- Define an intuitive steering law to
    - Correct heading error
    - Correct position error
    - Obey max steering angle bounds

Heading control law

- Combine three requirements



- Steer to align heading with desired heading (proportional to heading error)

$$\delta(t) = \psi(t)$$

- Steer to eliminate crosstrack error
    - Essentially proportional to error
    - Inversely proportional to speed
    - Limit effect for large errors with inverse tan
    - Gain k determined experimentally

$$\delta(t) = \tan^{-1}\left(\frac{ke(t)}{v_f(t)}\right)$$

- Maximum and minimum steering angle

$$\delta(t) \in [\delta_{min}, \delta_{max}]$$

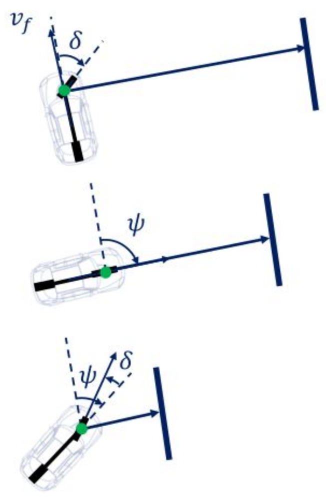
Combined steering law

- Stanley control law

$$\delta(t) = \psi(t) + \tan^{-1}\left(\frac{ke(t)}{v_f(t)}\right), \quad \delta(t) \in [\delta_{min}, \delta_{max}]$$

- For large heading error, steer in opposite direction
    - The larger the heading error, the larger the steering correction
    - Fixed at limit beyond maximum steering angle, assuming no crosstrack error
- For large positive crosstrack error

$$\tan^{-1}\left(\frac{ke(t)}{v_f(t)}\right) \approx \frac{\pi}{2} \;\rightarrow\; \delta(t) \approx \psi(t) + \frac{\pi}{2}$$

        - As heading changes due to steering angle, the heading correction counteracts the crosstrack correction, and drives the steering angle back to zero (This large value clamps the steering command to the
          maximum and the vehicle turns towards the path.)
        - The vehicle approaches the path, crosstrack error drops, and steering command starts to correct the heading alignment
        - 

Error dynamics

- The error dynamics when not at the maximum steering angle are

$$\dot{e}(t) = -v_f(t)\sin(\psi(t) - \delta(t)) = -v_f(t)\sin\left(\tan^{-1}\left(\frac{ke(t)}{v_f(t)}\right)\right)$$
$$= \frac{-ke(t)}{\sqrt{1+\left(\frac{ke(t)}{v_f}\right)^2}}$$

- For small crosstrack error, leads to exponential decay characteristics

$$\dot{e}(t) \approx -ke(t)$$

- The most interesting aspect of this investigation is that the decay rate is completely independent of the speed. So faster vehicles will travel farther while converging to the path, but will converge to the path at the same time as slower moving vehicles.

Adjustment

- Low speed operation
    - Inverse speed can cause numerical instability
    - Add softening constant to controller

$$\delta(t) = \psi(t) + \tan^{-1}\left(\frac{ke(t)}{k_s + v_f(t)}\right)$$

- Extra damping on heading
    - Avoid Stanley's response overly aggressive at high speeds,
    - Becomes an issue at higher speed in real vehicle
- Steer into constant radius curves
    - Improves tracking on curves by adding a feedforward term on heading

Stanley becomes a useful tool for moderate driving tasks as long as the vehicle avoids exiting the linear tire region.

More info about pure pursuit and Stanley controller:

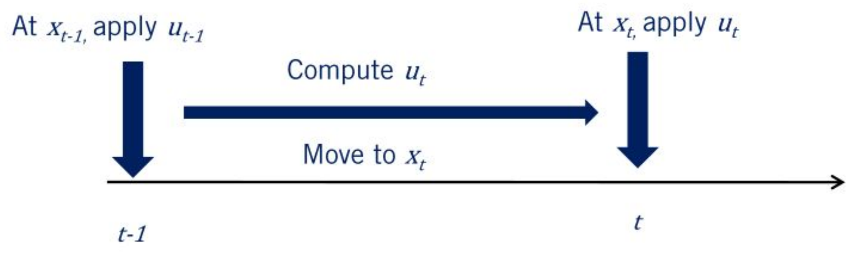
- Snider, J. M., "Automatic Steering Methods for Autonomous Automobile Path Tracking", Robotics Institute, Carnegie Mellon University, Pittsburg (February 2009).
- Hoffmann, G. et al., "Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing", Stanford University, (2007)

Model Predictive Control

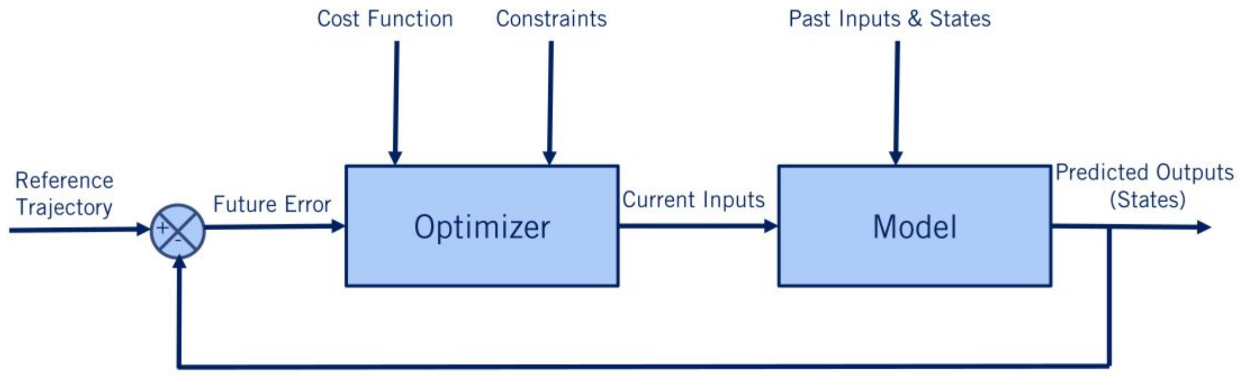
- Numerically solving an optimization problem at each time step
- Receding horizon approach
- Advantage:
    - Straightforward formulation
    - Explicitly handles constraints
    - Applicable to linear or nonlinear models
- Disadvantage
    - Computationally expensive

Receding horizon control

- Pick receding horizon length (T)
- For each time step t
- Set initial step to predict predict state x_t
    - Perform optimization over finite horizon t to T while traveling from x_{t-1} to x_t
    - Apply first control command u_t form optimization at time t



MPC block diagram



Linear MPC formulation

- Linear time-invariant discrete time model

$$x_{t+1} = Ax_t + Bu_t$$

- MPC seeks to find control policy U

$$U = \{u_{t|t}, u_{t+1|t}, u_{t+2|t}, \dots\}$$

- Objective function: regulation

$$J(x(t), U) = \sum_{j=t}^{t+T-1} x_{j|t}^T Q x_{j|t} + u_{j|t}^T R u_{j|t}$$

- Objective function: tracking

$$\delta x_{j|t} = x_{j|t,des} - x_{j|t} \qquad J(x(t), U) = \sum_{j=t}^{t+T-1} \delta x_{j|t}^T Q \delta x_{j|t} + u_{j|t}^T R u_{j|t}$$

Solution

- Unconstrained, finite horizon, discrete time problem formulation:

$$\min_{U \triangleq \{u_{t|t}, u_{t+1|t}, \dots\}} J(x(t), U) = x_{t+T|t}^{\mathrm{T}} Q_f \, x_{t+T|t} + \sum_{j=t}^{t+T-1} x_{j|t}^{\mathrm{T}} Q x_{j|t} + u_{j|t}^{\mathrm{T}} R u_{j|t}$$

$$s.t. \qquad x_{j+1|t} = A x_{t|t} + B u_{t|t} \,, \qquad t \le j \le t + T - 1$$

- Linear quadratic regulator, provides a closed form solution
  - Full state feedback: $u_t = -K x_t$
  - Control gain K is a matrix
  - Refer to supplemental materials

Nonlinear MPC formulation

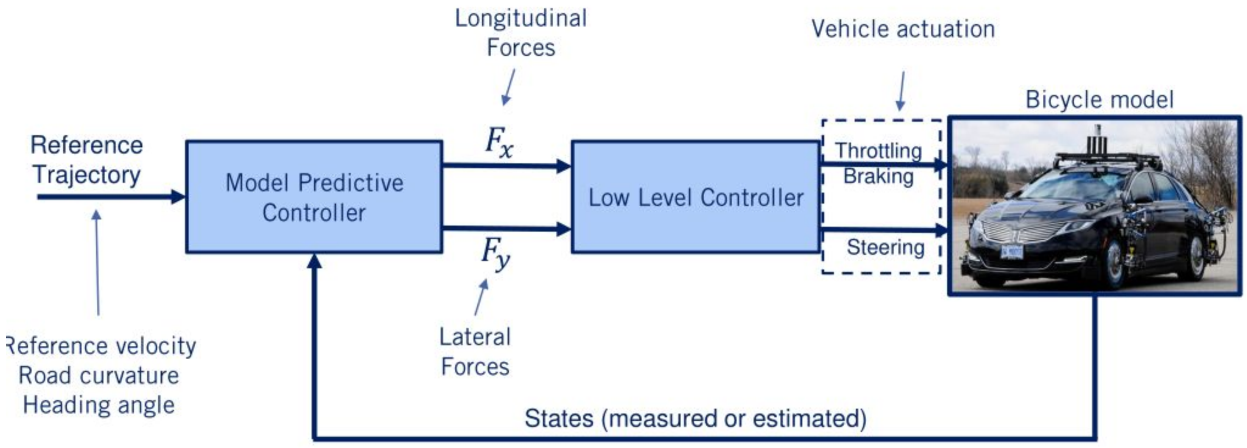
- Constrained nonlinear finite horizontal discrete time case

$$\min_{U \triangleq \{u_{t|t}, u_{t+1|t}, \dots\}} J(x(t), U) = \sum_{j=t}^{t+T} C(x_{j|t}, u_{j|t})$$

$$\begin{aligned} s.t. \qquad & x_{j+1|t} = f\left(x_{j|t}, u_{j|t}\right), && t \le j \le t + T - 1 \\ & x_{min} \le \; x_{j+1|t} \; \le x_{max} \,, && t \le j \le t + T - 1 \\ & u_{min} \le \; u_{j|t} \; \le u_{max} \,, && t \le j \le t + T - 1 \\ & g\left(x_{j|t}, u_{j|t}\right) \; \le 0, && t \le j \le t + T - 1 \\ & h\left(x_{j|t}, u_{j|t}\right) = 0, && t \le j \le t + T - 1 \end{aligned}$$

- No closed form solution, must be solved numerically

Vehicle lateral control



MPC

- Cost function -minimize
    - Deviation from desired trajectory
    - Minimization of control command magnitude
- Constraints - subjective to
    - Longitudinal and lateral dynamic model
    - These costs and constraints define the optimization used in our example, which then gets converted into actual vehicle commands by the low-level controller.
    - Tire force limits
- Can incorporate low-level controller, adding constraints for
    - Engine map
    - Full dynamic vehicle model
    - Actuator models
    - Tire force models

More about MPC

Falcone, P. et al., "Predictive Active Steering Control for Autonomous Vehicle Systems", IEEE (2007)