ROBOT DYNAMICS

By Samaksh Ujjawal

RRC, IIIT Hyderabad

BACKGROUND

Physics Prerequisites

- Forces & Torques: Drive robot movement, e.g. $\tau = F \cdot d$.
- Inertia: How robots resist changes in motion.
- Newton's Laws: Basis of motion, e.g. $F = m \cdot a$.

Kinematics

$$\text{Kinematics:Position}(\vec{p}),\text{Velocity}(\vec{v}),\text{Acceleration}(\vec{a})\text{Describe motion.}$$

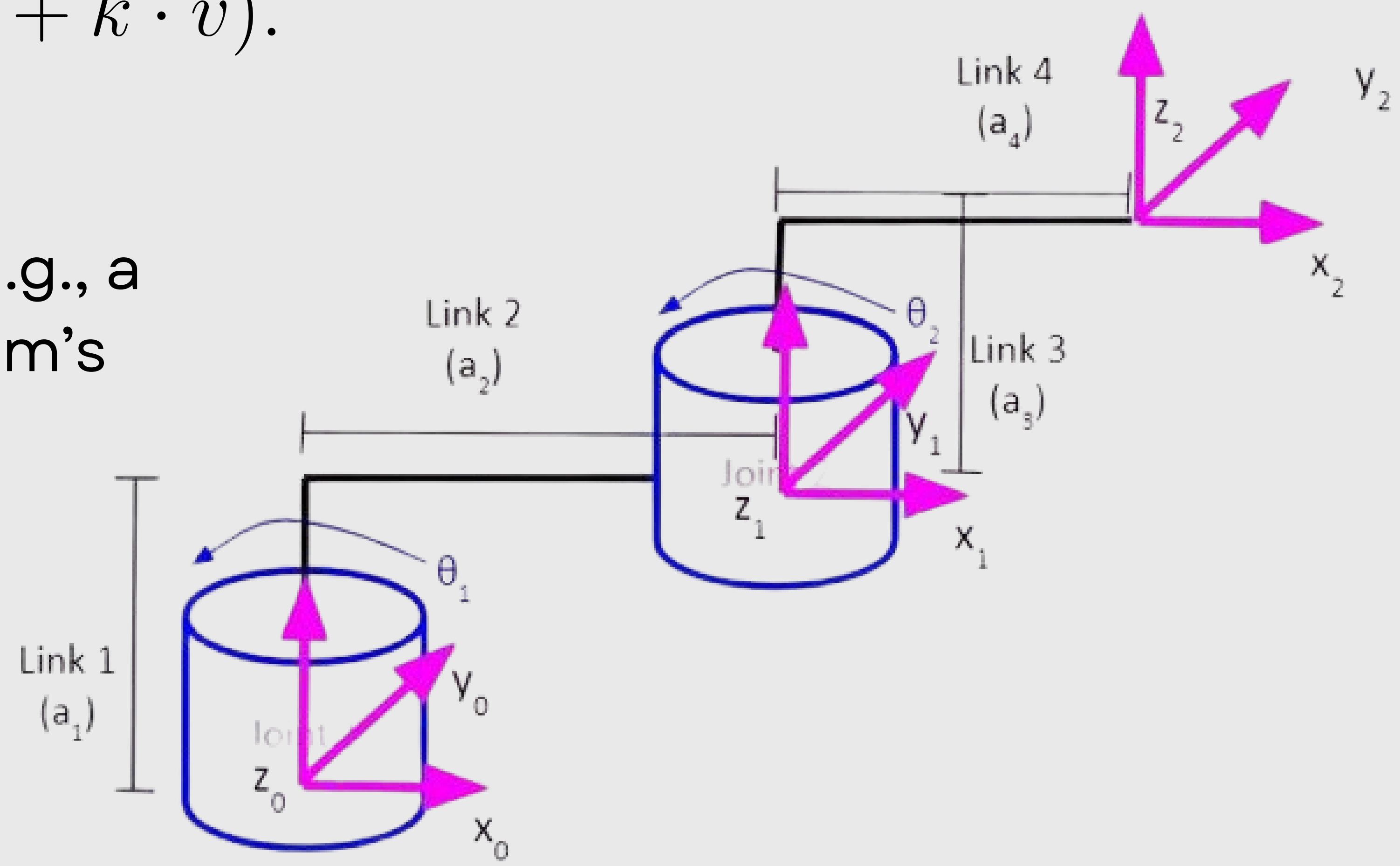
$$\text{Key Relationships:}\vec{v} = \frac{d\vec{p}}{dt}, \quad \vec{a} = \frac{d\vec{v}}{dt}.$$

Mathematics

- **Vectors**: Represent position, force, velocity. E.g., dot product calculates work; cross product computes torque.
- **Differential Equations**: Model dynamics. E.g., a drone's altitude follows

$$(\frac{d^2h}{dt^2} = -g + k \cdot v).$$

- **Matrices**: Transform coordinates. E.g., a rotation matrix adjusts a robotic arm's end-effector position.

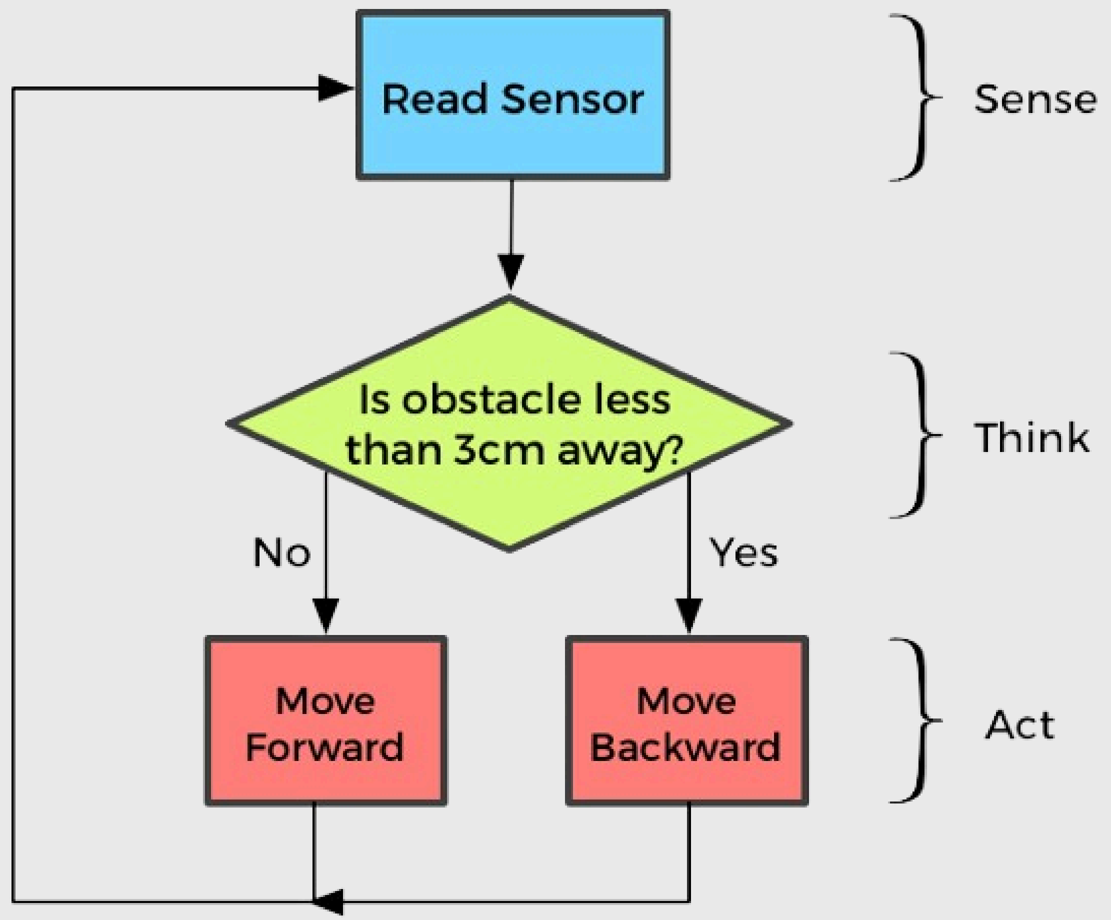


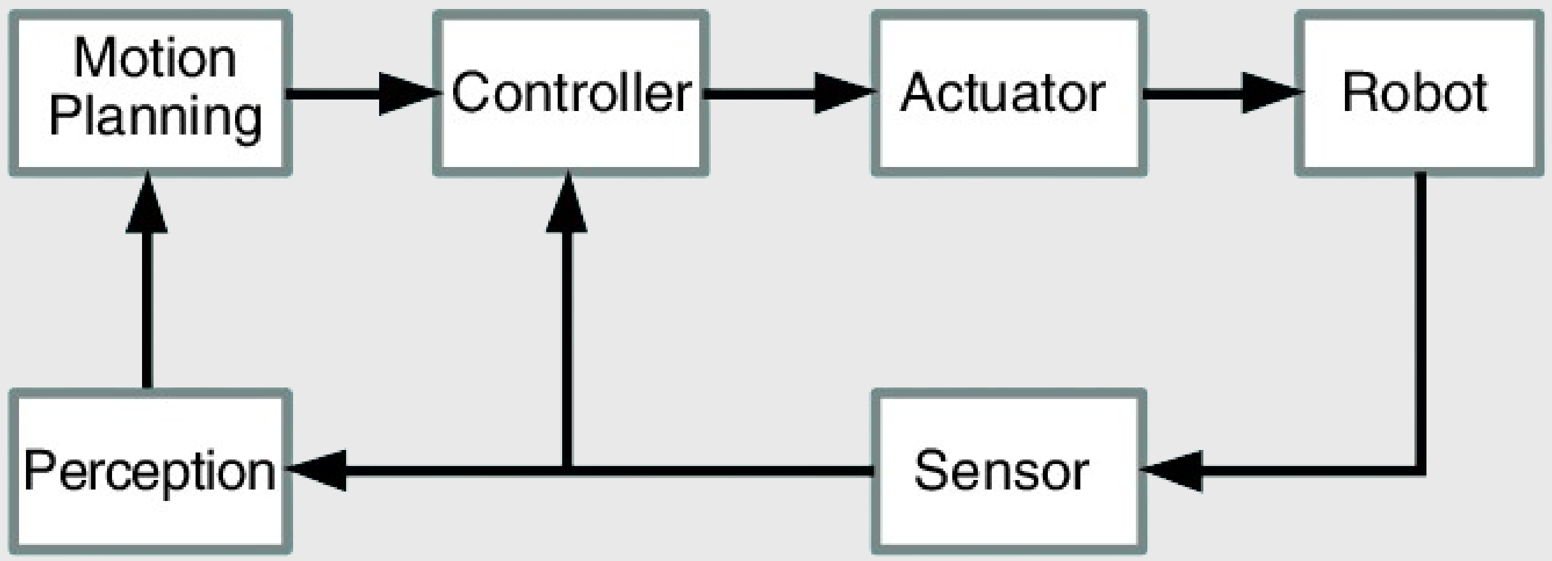
Robotics Basics

- **Definition**: Programmable machines performing tasks autonomously or semi-autonomously.
- **Key Components**:
  - Sensors: Detect environment (e.g., cameras, lidar, ultrasonic).
  - Actuators: Enable motion (e.g., motors, hydraulics, pneumatics).
  - Controllers: Process data and decide actions (e.g., microcontrollers, AI).
- **Types of Robots**:
  - Industrial: Arms for assembly or welding.
  - Mobile: Roombas or rovers for navigation.
  - Drones: Aerial tasks like delivery.
  - Humanoids: Assistance or research (e.g., Pepper).

Robotics Basics

- **How They Work:** Sense (e.g., detect obstacles), think (e.g., plan paths), act (e.g., move or grab).
- **Applications**: Industry (automation), healthcare (surgery), space (rovers), daily life (lawnmowers).
- **Example**: A Roomba senses walls, computes a path, and adjusts its motors to clean efficiently.





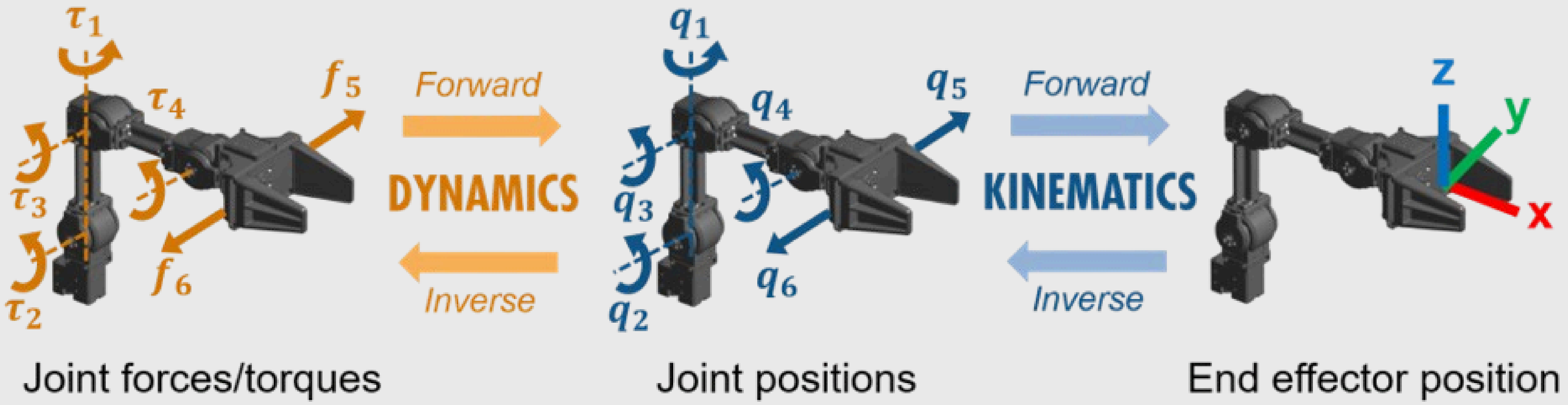
A typical robotics system

WHY STUDY ROBOT DYNAMICS?

- **Content**:
  - "In which direction are we going?"
  - "We are heading toward understanding the behavior of robots and controlling them."
  - "Understanding cause of motion --> Dynamics"
  - "While in motion --> Kinematics"
  - Dynamics enables us to predict and control how robots behave by analyzing forces and torques.
- **Purpose**: To build robots that move accurately and safely (e.g., a drone hovering or a robotic arm lifting).

KINEMATICS VS. DYNAMICS

- Kinematics: Study of motion without forces.
  - Focus: What is the motion? (Position, velocity, acceleration)
  - Example: "Where is the robot’s gripper based on joint angles?"
- Dynamics: Study of motion with forces.
  - Focus: Why does motion occur? (Forces, torques)
  - Example: "What torques move the gripper to that position?"
- Key Point: Dynamics builds on kinematics to control robot behavior.



WHAT IS ROBOT DYNAMICS?

- **Definition**: Study of how forces and torques cause motion in robots.
  - Significance:
    - Essential for controlling robot behavior (e.g., drones staying stable).
    - Enables simulation and design (e.g., robotic arms lifting objects).
  - Key Concepts:
    - Forward Dynamics: Forces → Motion.
    - Inverse Dynamics: Motion → Forces.
- **Example**: Calculating torques for a robotic arm to pick up a box.

Rigid-body equations of motion

$$\overbrace{\tau}^{\text{applied joint torque}} = \mathbf{M}(q)\underbrace{\ddot{q}}_{\text{joint acceleration}} + \mathbf{C}(q,\dot{q})\underbrace{\dot{q}}_{\text{joint velocity}} + g(q)$$

BASIC PHYSICS FOR ROBOT DYNAMICS

- **Newton's 2nd Law**: (F = m.a) (Force = mass × acceleration).

- **Torque**: Rotational force (T= F x d)).

- Robot Context:
  - Robots have links and joints with mass and inertia.
  - Forces and torques drive their motion.
- **Goal:** Use physics to understand and control motion.

NEWTON-EULER EQUATIONS – INTRODUCTION

What Are They?: Equations combining Newton's laws (linear) and Euler's laws (rotational) to model robot dynamics.

This method is jointly based on:

- Newton's 2nd Law of Motion Equation:

$$F = m_i \dot{v}_C$$

  and considering a 'rigid' link

- Euler's Angular Force/ Moment Equation:

$$N_{moment} = I_{CM_i} \dot{\omega}_i + \omega_i \times I_{CM_i} \omega_i$$

- **Purpose**: Calculate forces and torques for controlling robots (e.g., robotic arms).
- **Relevance**: Foundation for efficient algorithms like RNEA.

APPLYING NEWTON-EULER IN ROBOTICS

- RECURSIVE NEWTON-EULER ALGORITHM (RNEA):

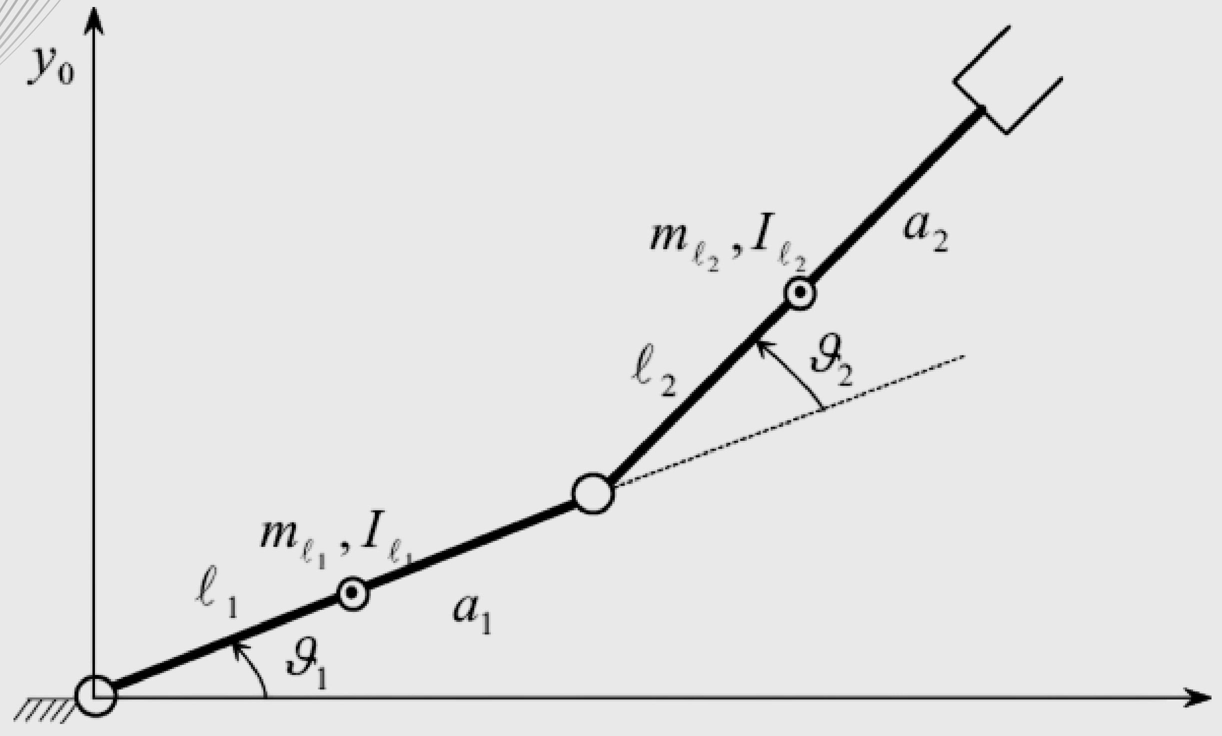
Forward Recursion:

- Compute position, velocity, acceleration of each link's center of mass.
- From base to end-effector.

Backward Recursion:

- Compute forces and torques on each link.
- From end-effector to base, calculate joint torques τ_1, τ_2, τ_3.

APPLYING NEWTON-EULER IN ROBOTICS



Backward dynamics

Inertial forces and moments

$$\mathbf{f}_i^* = -m_i \dot{\mathbf{v}}_{ci}$$

$$\mathbf{n}_i^* = -\mathbf{I}_i \dot{\boldsymbol{\omega}}_i - \boldsymbol{\omega}_i \times (\mathbf{I}_i \boldsymbol{\omega}_i)$$

Force and torque balance equations about the center of mass

$$\mathbf{f}_{i,i-1} = \mathbf{f}_{i+1,i} - m_i \mathbf{g} - \mathbf{f}_i^*$$

$$\mathbf{n}_{i,i-1} = \mathbf{n}_{i+1,i} + (\mathbf{r}_i + \mathbf{r}_{ci}) \times \mathbf{f}_{i,i-1} - \mathbf{r}_{ci} \times \mathbf{f}_{i+1,i} - \mathbf{n}_i^*$$

Torque in rotational joint

$$\boldsymbol{\tau}_i = {}^{i-1}\mathbf{n}_{i,i-1}^T \, {}^{i-1}\mathbf{z}_{i-1}$$

Forward kinematics

Angular velocity propagation

$${}^{i}\mathbf{R}_i = \begin{bmatrix} \cos(\theta_i) & \sin(\theta_i) & 0 \\ -\cos(\alpha_i)\sin(\theta_i) & \cos(\alpha_i)\cos(\theta_i) & \sin(\alpha_i) \\ \sin(\alpha_i)\sin(\theta_i) & -\sin(\alpha_i)\cos(\theta_i) & \cos(\alpha_i) \end{bmatrix}$$

$${}^{i-1}\mathbf{Z}_{i-1} = [0 \quad 0 \quad 1]^T$$

Angular acceleration propagation

$$\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1} + \mathbf{z}_{i-1}\ddot{\theta}_i + \boldsymbol{\omega}_{i-1} + \mathbf{z}_{i-1}\dot{\theta}_i$$

$${}^{i}\dot{\boldsymbol{\omega}}_i = {}^{i}\boldsymbol{R}_{i-1}\left({}^{i-1}\dot{\boldsymbol{\omega}}_{i-1} + {}^{i-1}\mathbf{z}_{i-1}\ddot{\theta}_i + {}^{i-1}\boldsymbol{\omega}_{i-1} + {}^{i-1}\mathbf{z}_{i-1}\dot{\theta}_i\right)$$

Linear velocity propagation

$$\mathbf{v}_i = \mathbf{v}_{i-1} + \boldsymbol{\omega}_i \times \mathbf{r}_i$$

$${}^{i}\mathbf{v}_i = {}^{i}\boldsymbol{R}_{i-1} \, {}^{i-1}\mathbf{v}_{i-1} + {}^{i}\boldsymbol{\omega}_i \times {}^{i}\mathbf{r}_i$$

$${}^{i}\mathbf{r}_i = [a_i \quad d_i \sin(\alpha_i) \quad d_i \cos(\alpha_i)]^T$$

Linear acceleration propagation

$$\dot{\mathbf{v}}_i = \dot{\mathbf{v}}_{i-1} + \dot{\boldsymbol{\omega}}_i \times \mathbf{r}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{r}_i)$$

$${}^{i}\dot{\mathbf{v}}_i = {}^{i}\boldsymbol{R}_{i-1} \, {}^{i-1}\dot{\mathbf{v}}_{i-1} + {}^{i}\dot{\boldsymbol{\omega}}_i \times {}^{i}\mathbf{r}_i + {}^{i}\boldsymbol{\omega}_i \times ({}^{i}\boldsymbol{\omega}_i \times {}^{i}\mathbf{r}_i)$$

Linear acceleration of the center of mass

$$\dot{\mathbf{v}}_{ci} = \dot{\mathbf{v}}_i + \dot{\boldsymbol{\omega}}_i + \dot{\boldsymbol{\omega}}_i \times \mathbf{r}_{ci} + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{r}_{ci})$$

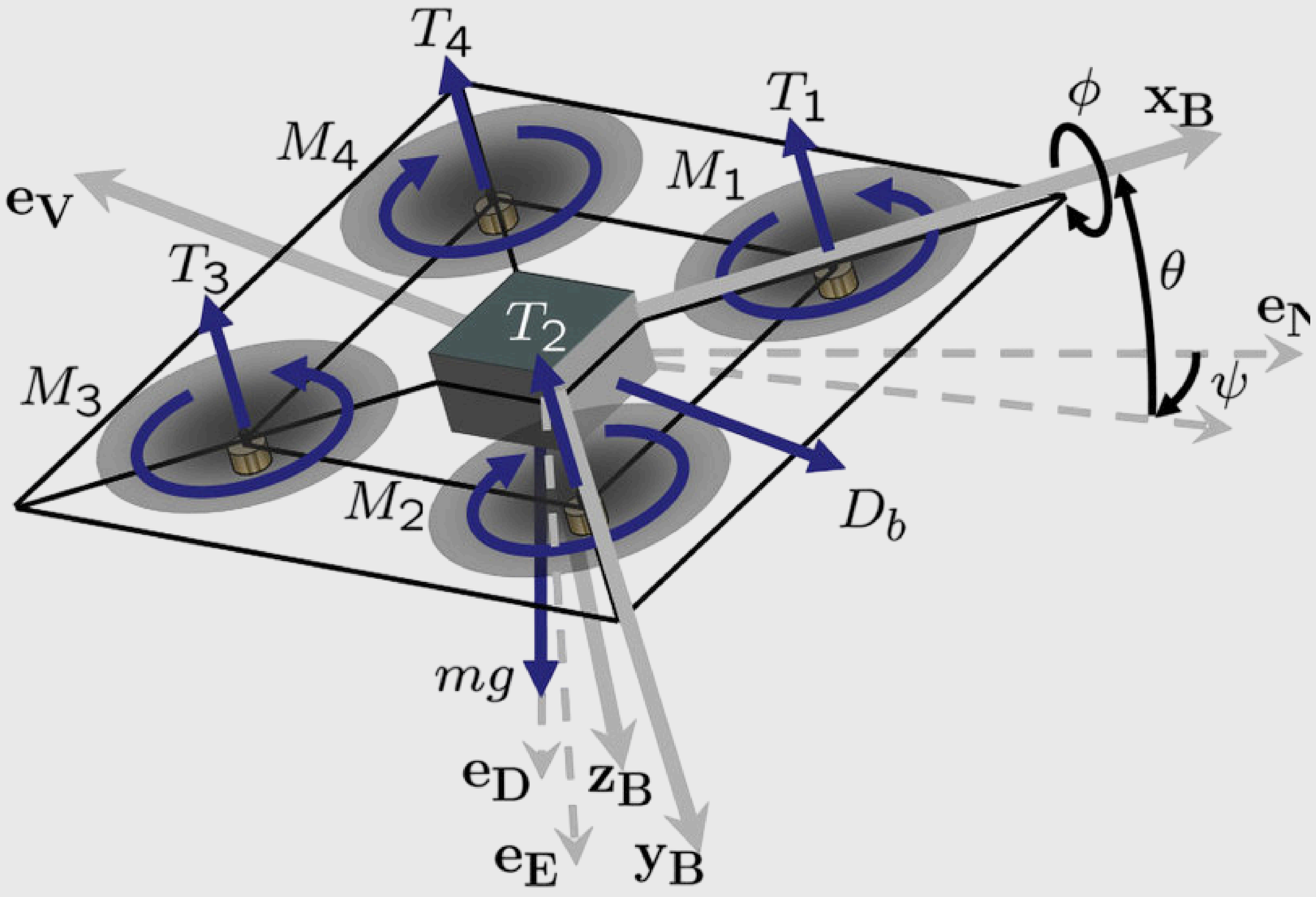
$${}^{i}\dot{\mathbf{v}}_{ci} = {}^{i}\dot{\mathbf{v}}_i + {}^{i}\dot{\boldsymbol{\omega}}_i + {}^{i}\dot{\boldsymbol{\omega}}_i \times {}^{i}\mathbf{r}_{ci} + {}^{i}\boldsymbol{\omega}_i \times ({}^{i}\boldsymbol{\omega}_i \times {}^{i}\mathbf{r}_{ci})$$

Acceleration of gravity

$${}^{i}\mathbf{g} = {}^{i}\mathbf{R}_{i-1} \, {}^{i-1}\mathbf{g}$$

QUADROTOR DYNAMICS – INTRODUCTION

- **What's a Quadrotor?**:
  - Drone with four rotors.
  - Dynamics Role: Applies forces (rotor thrust) to control behavior and stability.
  - Key Forces: Thrust, gravity, drag.



QUADROTOR DYNAMICS – THE BASICS

- Degrees of Freedom
    - A quadrotor, or quadcopter, has six degrees of freedom, which means it can move in six different ways:
    - **3 Translational**: These are movements in space:
        - Up and down (changing altitude).
        - Left and right (sideways motion).
        - Forward and backward.
    - **3 Rotational:** These are rotations around its axes:
        - Roll: Tilting side to side (like leaning left or right).
        - Pitch: Tilting forward or backward.
        - Yaw: Turning left or right (like spinning in place).

QUADROTOR DYNAMICS – THE BASICS

How Control Works

- Thrust and Torque are changed by changing the speeds of the rotors, the quadrotor adjust its roll, pitch, yaw, and altitude.

1. Roll (Tilting Side to Side)

  - What it does: Roll tilts the quadrotor left or right.
  - How it's controlled: To roll right, the quadrotor speeds up the left rotors and slows down the right rotors. This creates more lift on the left side, tilting it to the right.
- Example: Imagine the quadrotor has four rotors:
- Rotor 1 (front-right), Rotor 2 (front-left), Rotor 3 (back-left), Rotor 4 (back-right).
- To roll right:
  - Speed up rotors 2 and 3 (left side).
  - Slow down rotors 1 and 4 (right side).

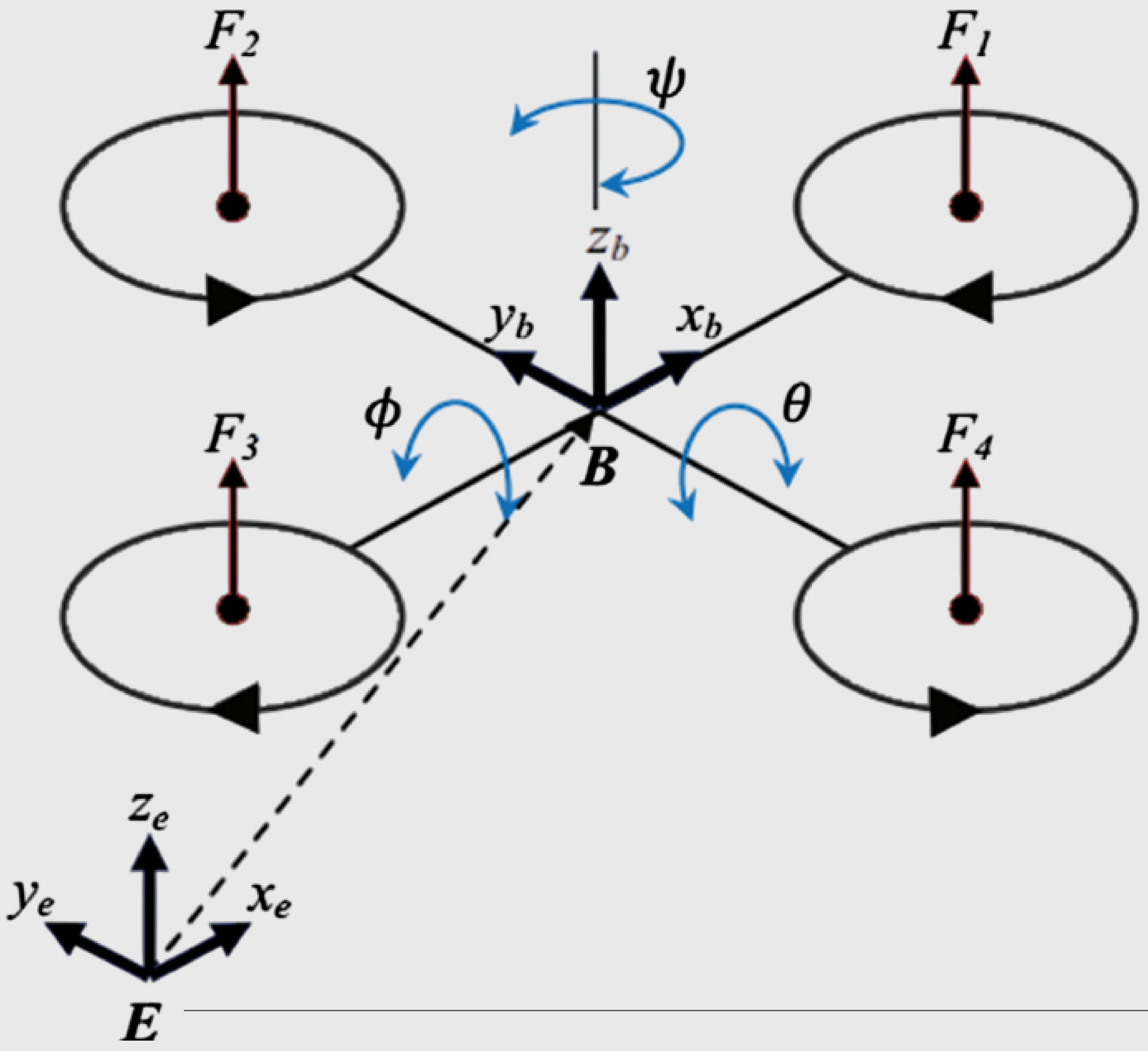
Result: More lift on the left tilts the quadrotor to the right.

QUADROTOR DYNAMICS – THE BASICS

Pitch (Tilting Forward or Backward)

- What it does: Pitch tilts the quadrotor forward or backward.
- How it's controlled: To pitch forward, the quadrotor speeds up the back rotors and slows down the front rotors. This creates more lift at the back, tilting it forward.
- Example: Using the same rotor setup:
  - To pitch forward:
    - Speed up rotors 3 and 4 (back).
    - Slow down rotors 1 and 2 (front).
  - Result: More lift at the back tilts the quadrotor forward.

QUADROTOR DYNAMICS – THE BASICS



QUADROTOR DYNAMICS – THE BASICS

Yaw (Turning Left or Right)

- What it does: Yaw rotates the quadrotor around its vertical axis, like spinning in place.
- How it's controlled: To yaw clockwise, the quadrotor adjusts rotor speeds to create a net twisting force (torque). Rotors spinning clockwise and counterclockwise work together.
    - Suppose rotors 1 and 4 spin clockwise, while rotors 2 and 3 spin counterclockwise.
    - To yaw clockwise:
        - Speed up rotors 1 and 4 (clockwise torque).
        - Slow down rotors 2 and 3 (counterclockwise torque).
- Result: A net clockwise torque spins the quadrotor clockwise.

QUADROTOR DYNAMICS – THE BASICS

Altitude (Moving Up or Down)

- To go up, all four rotors speed up equally, increasing total thrust.
- To go down, all four rotors slow down equally, reducing total thrust.

Translational Motion (Moving in a Direction)

Roll, pitch, and yaw control the quadrotor's orientation, but they also help it move in space. For example:

- To move forward, it pitches forward slightly. This tilts the thrust, giving a forward push.
- To move sideways, it rolls to one side, tilting the thrust in that direction.

QUADROTOR DYNAMICS – SIMPLIFIED MODEL

$V_B = [u,\ v,\ w]^T$, $\Omega = [p,\ q,\ r]^T$, $F = [F_{XB},\ F_{YB},\ F_{ZB}]^T$.

$$(\frac{dV_B}{dt})_I = (\frac{dV_B}{dt})_B + \Omega \times V_B = \frac{F}{m}$$

$$\dot{u} = rv - qw + \frac{F_{XB}}{m}$$

$$\dot{v} = pw - ru + \frac{F_{YB}}{m}$$

$$\dot{w} = qu - pv + \frac{F_{ZB}}{m}$$

Translational Dynamics

NB: Unlike mass, moment of inertia is not a scalar.

$$J = \begin{pmatrix} J_{xx} & J_{xy} & J_{xz} \\ J_{yx} & J_{yy} & J_{yz} \\ J_{zx} & J_{zy} & J_{zz} \end{pmatrix}$$

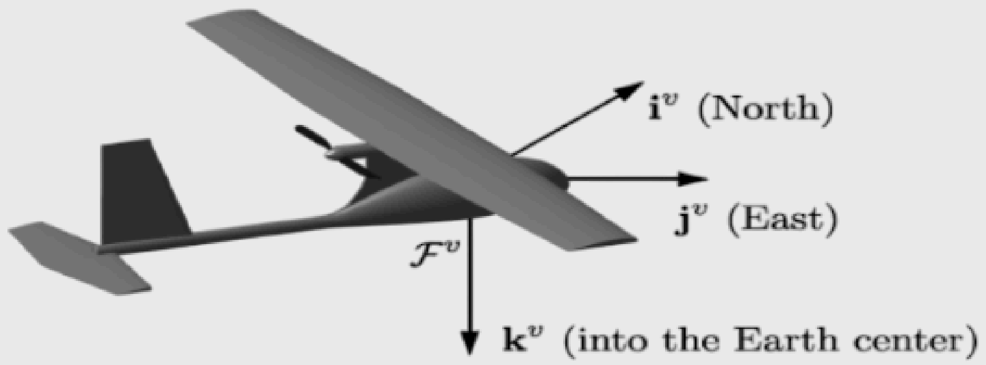
$$J(\frac{d\Omega}{dt})_I = J(\frac{d\Omega}{dt})_B + \Omega \times J\Omega = \tau$$

$J_{xy} = J_{yx},\ J_{xz} = J_{zx},\ J_{zy} = J_{yz}$.
Generally, J_{xy} and J_{yz} magnitude is small when compared to the others and can be neglected. For quadrotor $J_{xz} = 0$ and for fixed wing aircraft $J_{xz} \neq 0$.

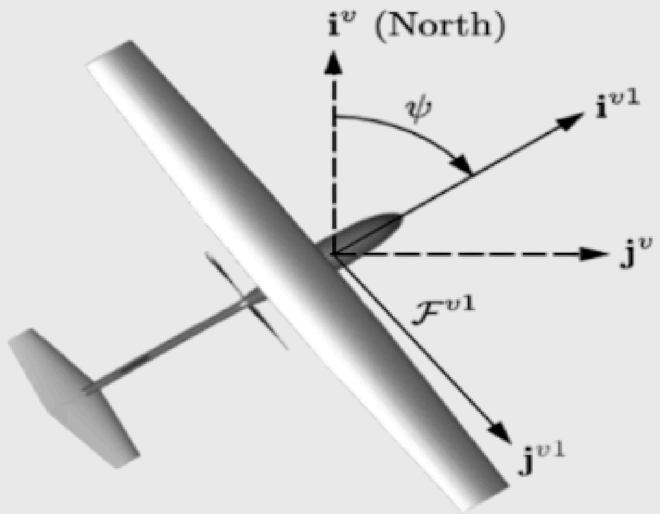
Rotational Dynamics

FIXED-WING DYNAMICS – INTRODUCTION

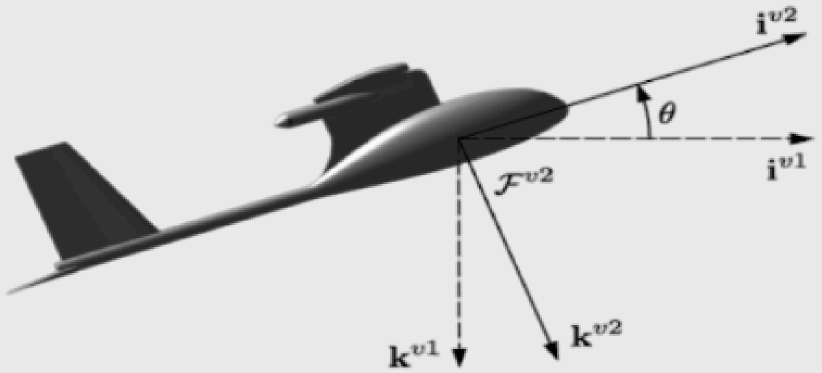
- What's a Fixed-Wing Robot?: Airplane-like robot with fixed wings.
- Dynamics Role: Balances lift, thrust, drag, gravity for flight control.
- Key Forces:
  - Lift
  - Thrust
  - Drag
  - Gravity.



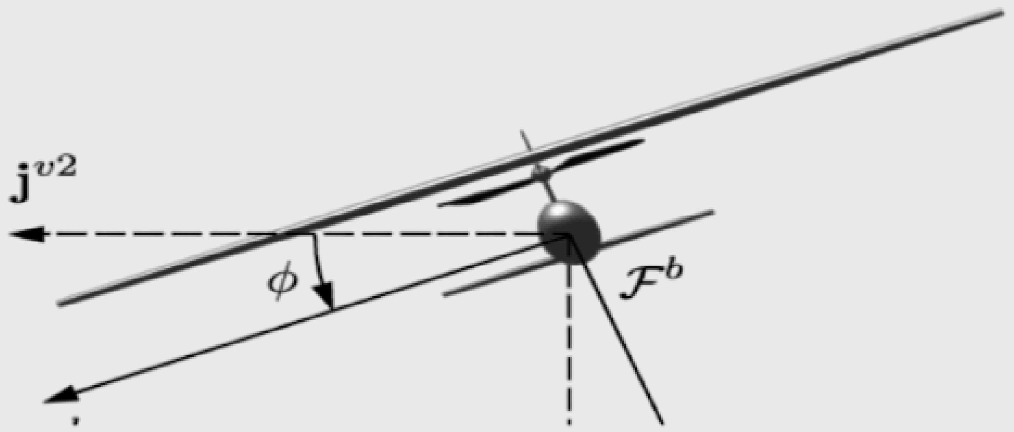
a) Inertial frame



b) Yaw angle



c) Pitch angle



d) Roll angle

FIXED-WING DYNAMICS – BASICS

Inertial Frame of Reference

Position: x, y, z. Velocity: v_x, v_y, v_z

Acceleration: a_x, a_y, a_z

Body Frame of Reference

Velocity: u, v, w

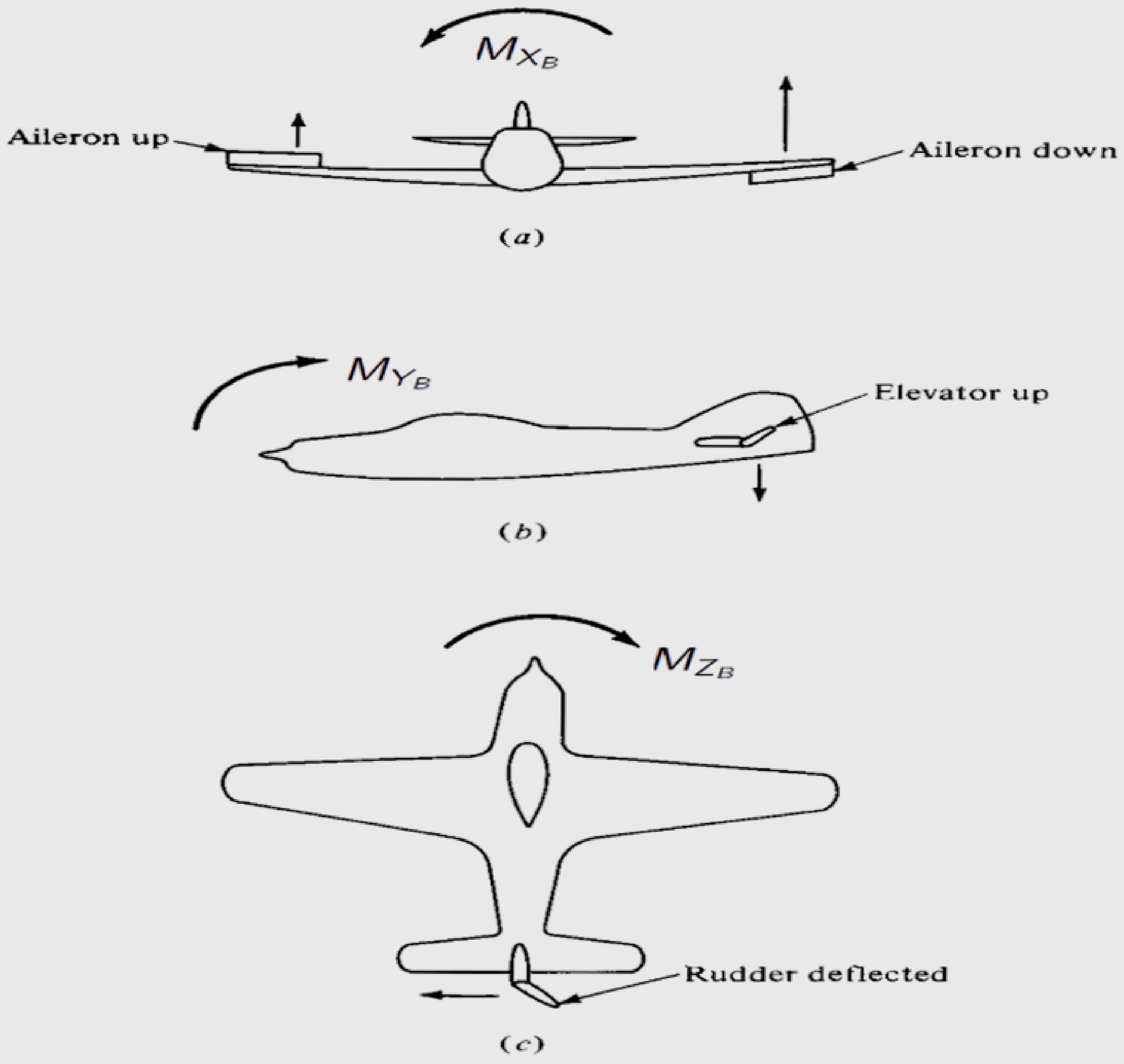
Acceleration: $\dot{u} = \frac{du}{dt}$, $\dot{v} = \frac{dv}{dt}$, $\dot{w} = \frac{dw}{dt}$

Angular velocity: p, q, r

Orientation

Euler angles: ψ, θ, ϕ

Remark: Altitude, $h = -z$.



FIXED-WING DYNAMICS – BASICS

Applying equation of Coriolis

Let $\bar{V}_B = [u,\ v,\ w]^T$ and $\bar{\Omega}_B = [p,\ q,\ r]^T$

$$(\dot{\bar{V}}_B)_I = (\dot{\bar{V}}_B)_B + \bar{\Omega}_B \times \bar{V}_B = \frac{\bar{F}}{m}$$

$\bar{F} = [F_{X_B},\ F_{Y_B},\ F_{Z_B}]^T = \bar{F}_g + \bar{F}_a + \bar{F}_t$
$\bar{F}_g$: force due to gravity, $\bar{F}_a$: force due to aerodynamic effects, $\bar{F}_t$: force due to propeller thrust.

Translational Dynamics

$\bar{\Omega}_B = [p,\ q,\ r]^T$

$$\bar{J}(\dot{\bar{\Omega}}_B)_I = \bar{J}(\dot{\bar{\Omega}}_B)_B + \bar{\Omega}_B \times \bar{J}\bar{\Omega}_B = \bar{M}$$

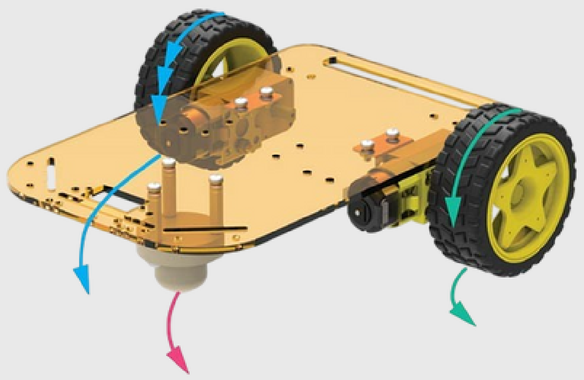
$$\bar{J} = \begin{pmatrix} J_{xx} & J_{xy} & J_{xz} \\ J_{yx} & J_{yy} & J_{yz} \\ J_{zx} & J_{zy} & J_{zz} \end{pmatrix}$$

$\bar{M} = [M_{X_B},\ M_{Y_B},\ M_{Z_B}]^T$

Rotational Dynamics

GROUND-VEHICLE DYNAMICS – INTRODUCTION

- Wheeled Robots
  - Differential-drive: Two independently driven wheels (plus optional casters).
  - Ackermann (Car-like): Front-steering geometry, suitable for higher speeds (e.g., autonomous cars).
  - Omni-directional: Mecanum or omni-wheels allow movement in any planar direction without re-orienting.



Differential-drive

Ackermann

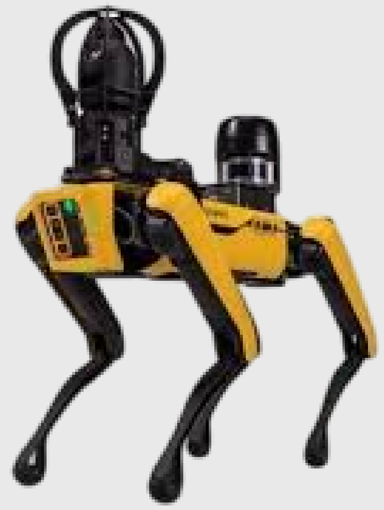
Omni-directional

GROUND-VEHICLE DYNAMICS – INTRODUCTION

- Tracked Robots
  - Continuous tracks (like tanks): high traction on rough terrain, but typically slower and less energy-efficient than wheels.
- Legged Robots
  - Bipeds (e.g., humanoids)
  - Quadrupeds (e.g., Boston Dynamics' Spot)
  - Hexapods, Octopods, etc.: increased stability and obstacle negotiation.

Differential-drive

Biped



Quadrupeds

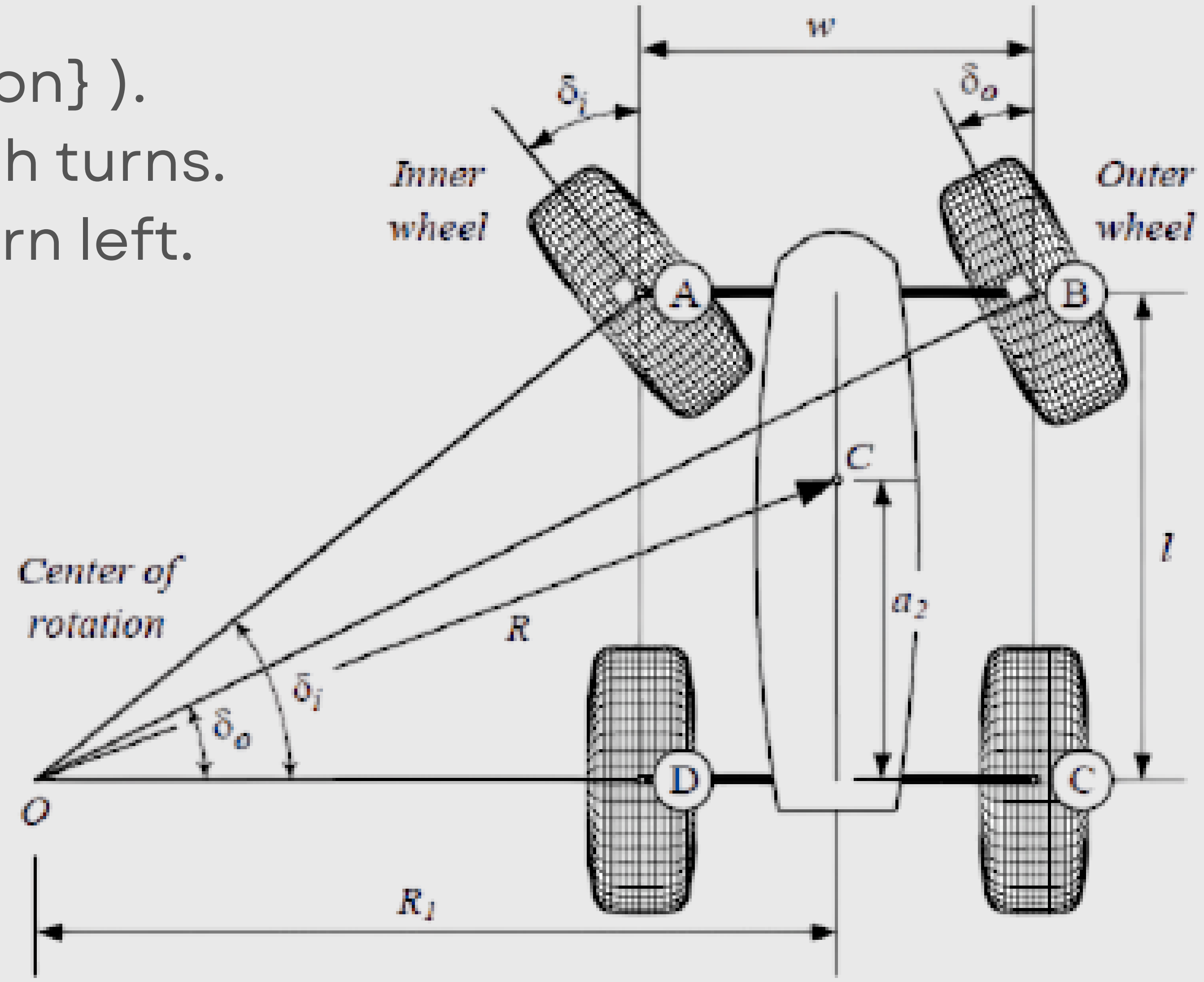
Hexapods

GROUND-VEHICLE DYNAMICS – INTRODUCTION

- Hybrid/Modular
  - Combinations (e.g., wheeled-legged robots that switch modes for speed vs. rough terrain).
- Key Forces: Traction, friction, gravity.
- Visual: Image of a rover with traction and friction force arrows.

GROUND-VEHICLE DYNAMICS – BASICS

- Motion Types: Straight-line (acceleration), turning (steering).
- Model: (F = m.a = F_{traction} - F_{friction}).
- Turning: Ackermann steering for smooth turns.
  - Example: Adjust steering angle → Turn left.

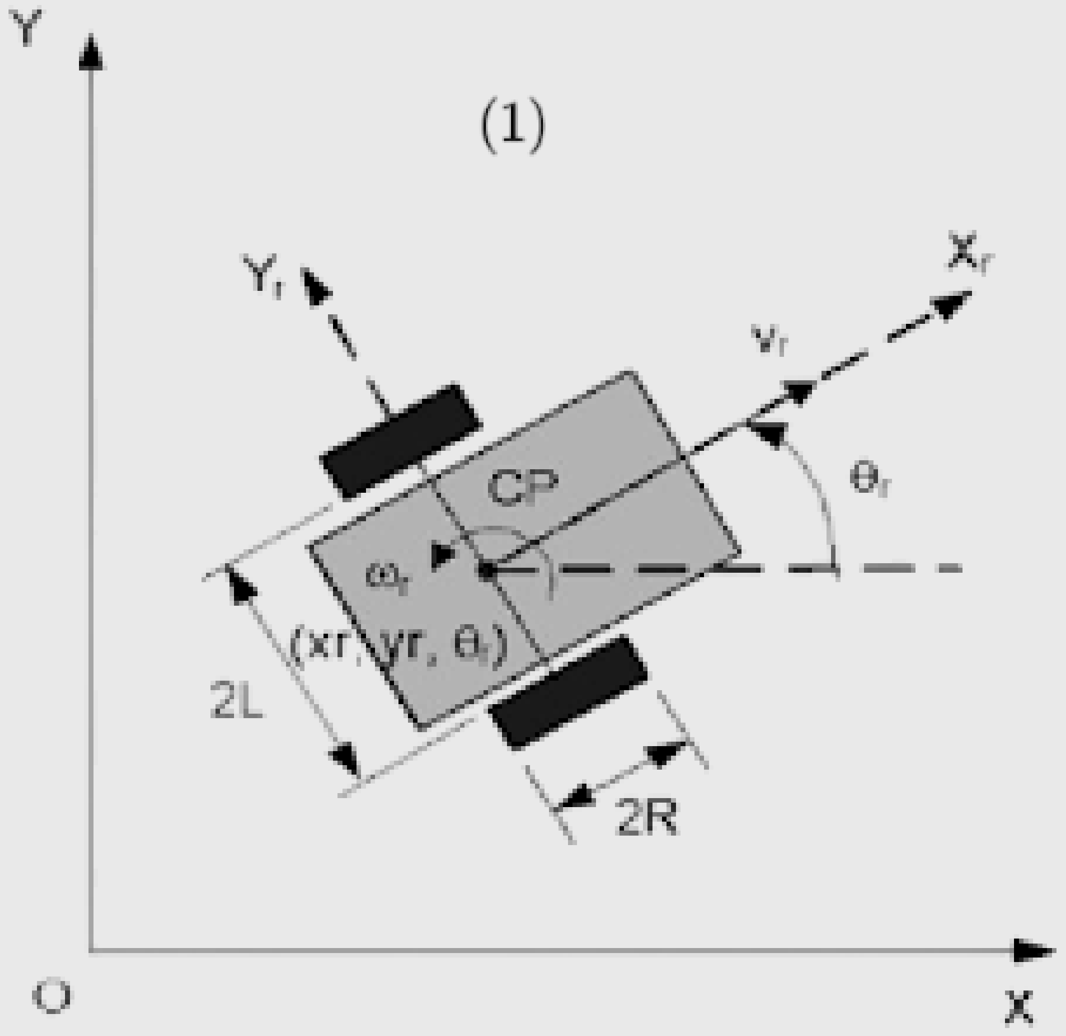


GROUND-VEHICLE DYNAMICS – BASICS

Differential Drive Vehicles

The dynamic model for differential drive vehicles is based on the conservation of linear and angular momentum. The equations are given as follows:

$$\begin{cases} m \cdot \dot{v}_x = F_{xr} + F_{xl}\cos(\phi) \\ m \cdot \dot{v}_y = F_{yl} + F_{yr}\sin(\phi) \\ I_z \cdot \dot{\omega} = (F_{xr} - F_{xl}) \cdot r + (F_{yl} - F_{yr}) \cdot r \end{cases} \quad (1)$$



where:

- m: mass of the robot (kg),
- I_z: moment of inertia about the vertical axis (kg · m²),
- v_x, v_y: linear velocities in x and y directions (m/s),
- ω: angular velocity (yaw rate) (rad/s),
- F_{xl}, F_{xr}: longitudinal forces on the left and right wheels (N),
- F_{yl}, F_{yr}: lateral forces (N),
- r: wheel radius (m),
- ϕ: steering angle (rad), typically 0 for differential drive.

GROUND-VEHICLE DYNAMICS – BASICS

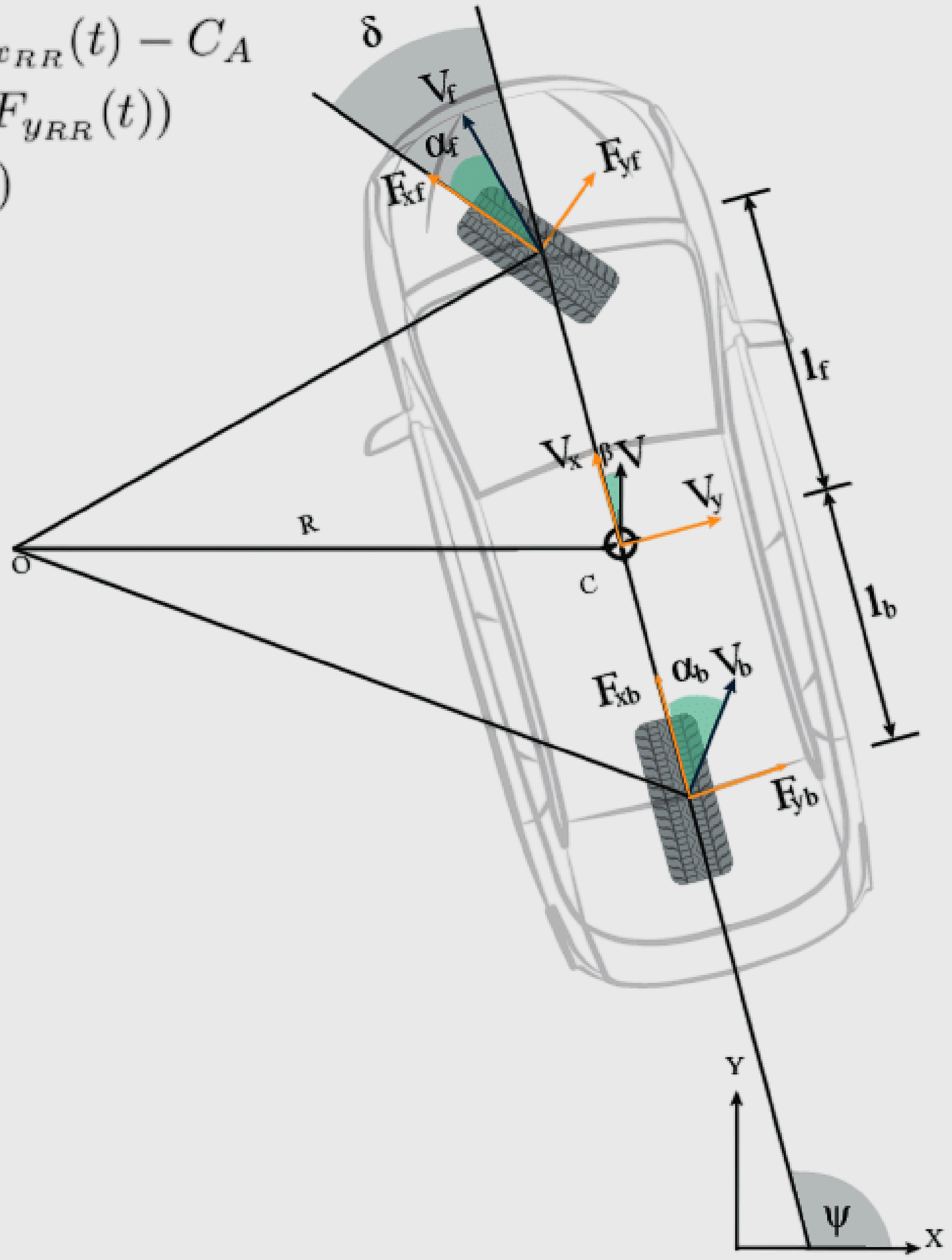
Car-Like Vehicles (Bicycle Model)

The dynamic model for car-like vehicles is often represented using the bicycle model, simplifying the vehicle to a single front and rear wheel. The governing differential equations are:

$$\begin{cases} \frac{d}{dt}v_x(t) = v_y(t)\cdot r(t) + \frac{1}{m}\left((F_{x_{FL}}(t)+F_{x_{FR}}(t))\cdot\cos(\delta(t)) - (F_{y_{FL}}(t)+F_{y_{FR}}(t))\cdot\sin(\delta(t)) + F_{x_{RL}}(t)+F_{x_{RR}}(t) - C_A \right. \\ \frac{d}{dt}v_y(t) = -v_x(t)\cdot r(t) + \frac{1}{m}\left((F_{x_{FL}}(t)+F_{x_{FR}}(t))\cdot\sin(\delta(t)) + (F_{y_{FL}}(t)+F_{y_{FR}}(t))\cdot\cos(\delta(t)) + F_{y_{RL}}(t)+F_{y_{RR}}(t)\right) \\ \frac{d}{dt}r(t) = \frac{1}{J}\left(a\cdot((F_{x_{FL}}(t)+F_{x_{FR}}(t))\cdot\sin(\delta(t)) + (F_{y_{FL}}(t)+F_{y_{FR}}(t))\cdot\cos(\delta(t))) - b\cdot(F_{y_{RL}}(t)+F_{y_{RR}}(t))\right) \end{cases} \tag{3}$$

where:

- m: vehicle mass (kg),
- $J = \frac{1}{(0.5\cdot(a+b))^2\cdot m}$: moment of inertia (kg $\cdot$ m^2),
- a, b: distances from the center of gravity to front and rear axles (m),
- $F_{x_i}(t), F_{y_i}(t)$: longitudinal and lateral tire forces for each wheel ($i = \{FL, FR, RL, RR\}$) (N),
- $\delta(t)$: steering angle (rad),
- C_A: air resistance coefficient (1/m),
- $v_x(t), v_y(t)$: longitudinal and lateral velocities (m/s),
- $r(t)$: yaw rate (rad/s).



THANK YOU