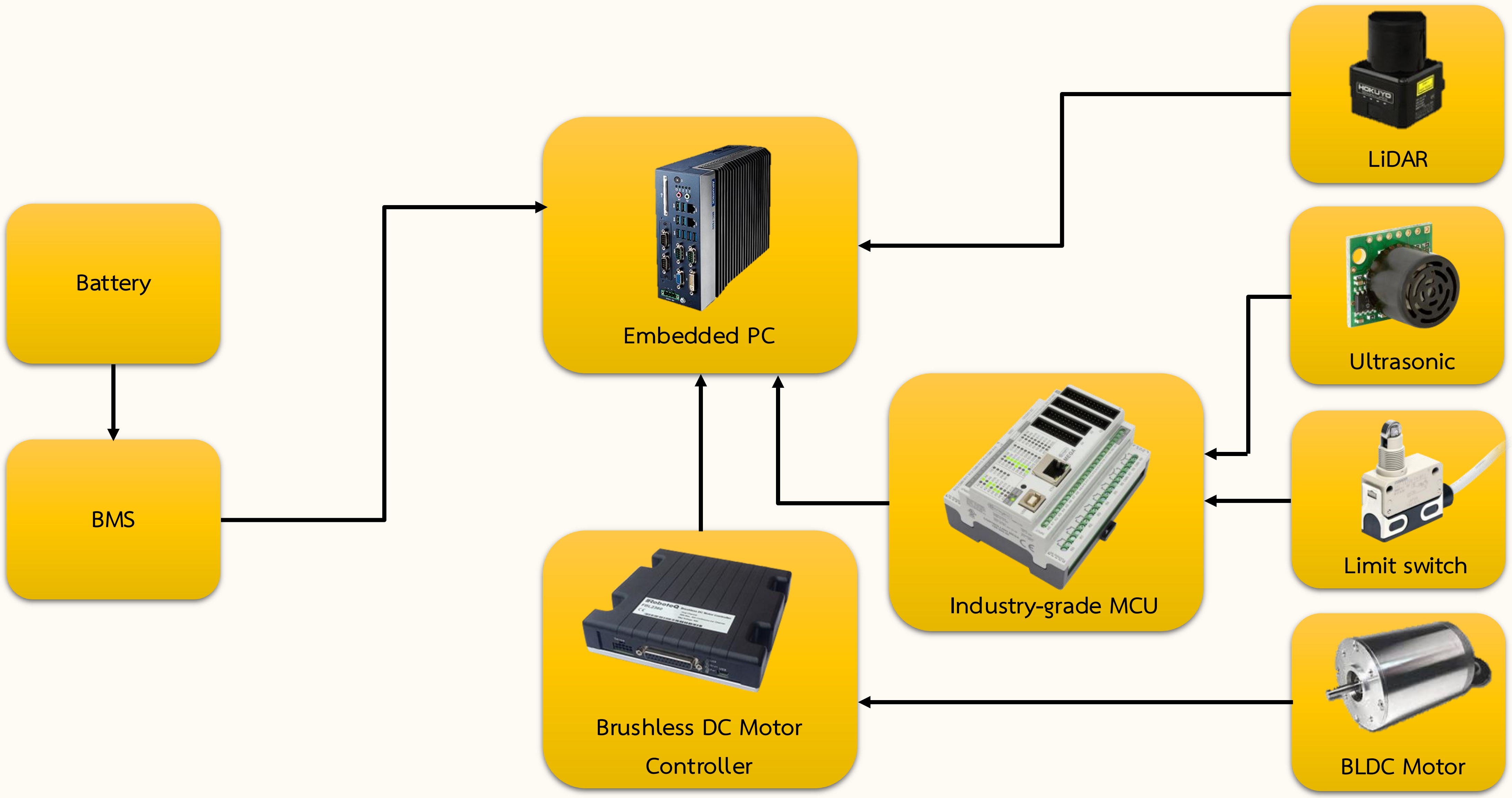
Dr. Sarucha Yanyong
Department of Robotics and AI Engineering, School of Engineering, KMITL

MOBILE ROBOT KINEMATICS 101

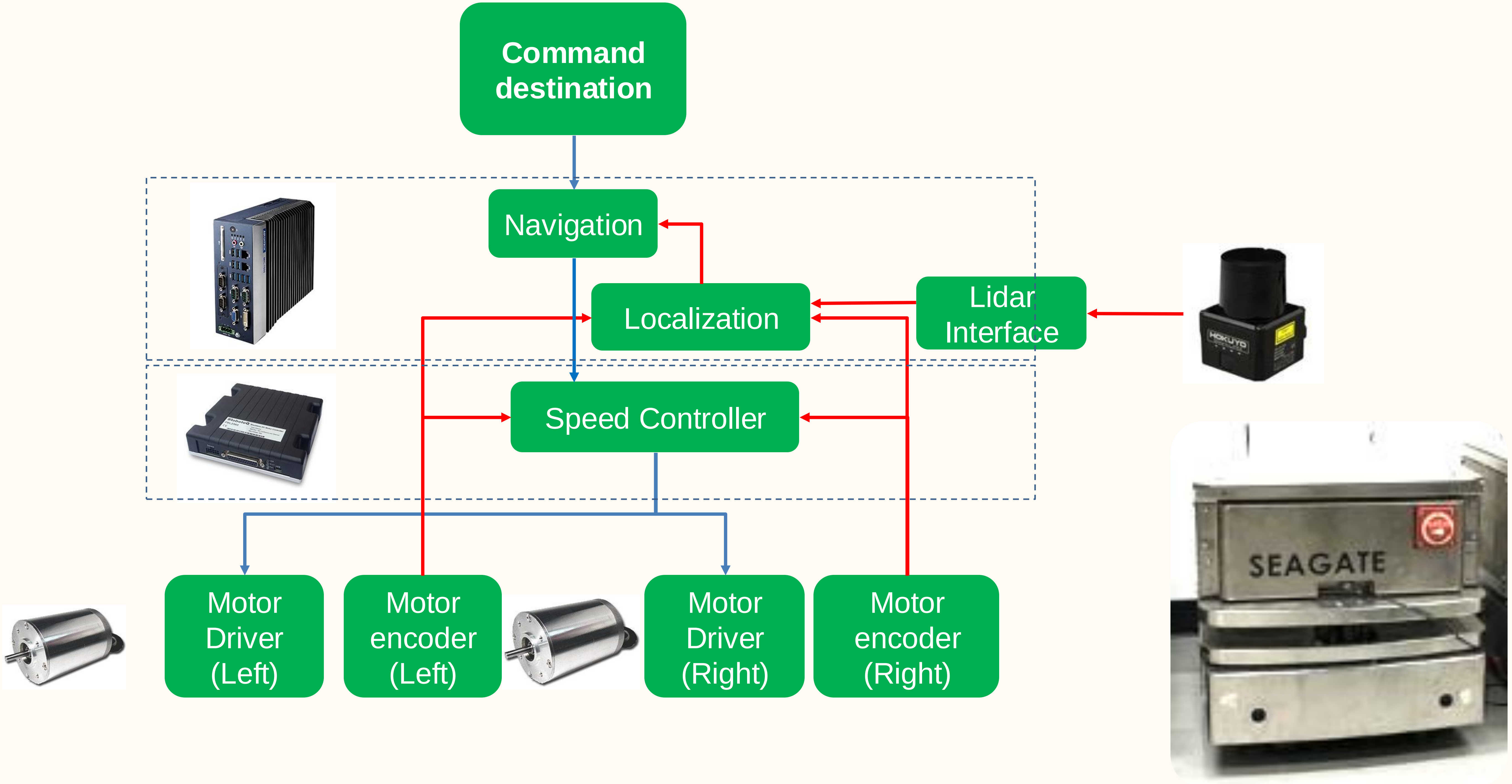
Hardware Connection

Short-review



Navigation Workflow

Short-brief



Dealing with low-level

Forward/Inverse Kinematics are needed!!

Overview of Mobile Robot Kinematics

Kinematics = The branch of mechanics that deals with the motion of objects without considering the forces that cause the motion.

Used to describe the motion of robots, including their position, velocity, and acceleration.

Overview of Mobile Robot Kinematics (2)

Kinematics = Mathematical framework to translate desired robot motion (e.g., moving forward, turning) into commands for the robot's actuators (e.g., wheel velocities).

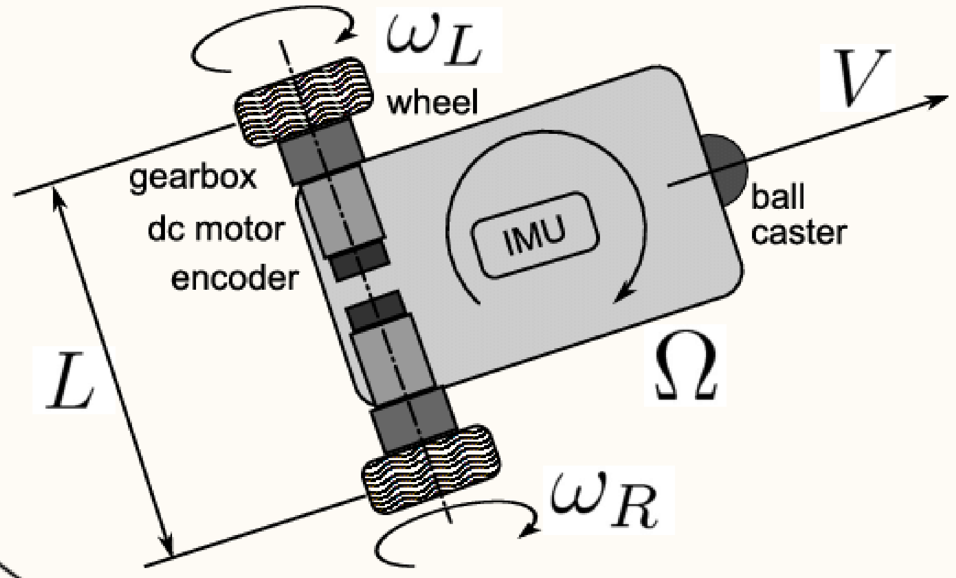
Overview of Mobile Robot Kinematics (3)

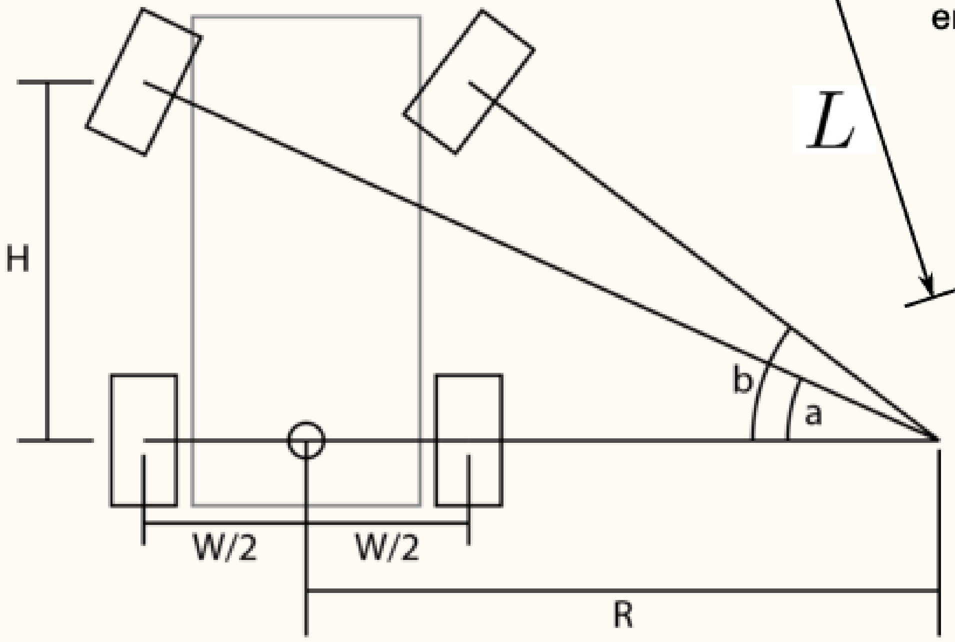
Types of Mobile Robot Kinematics

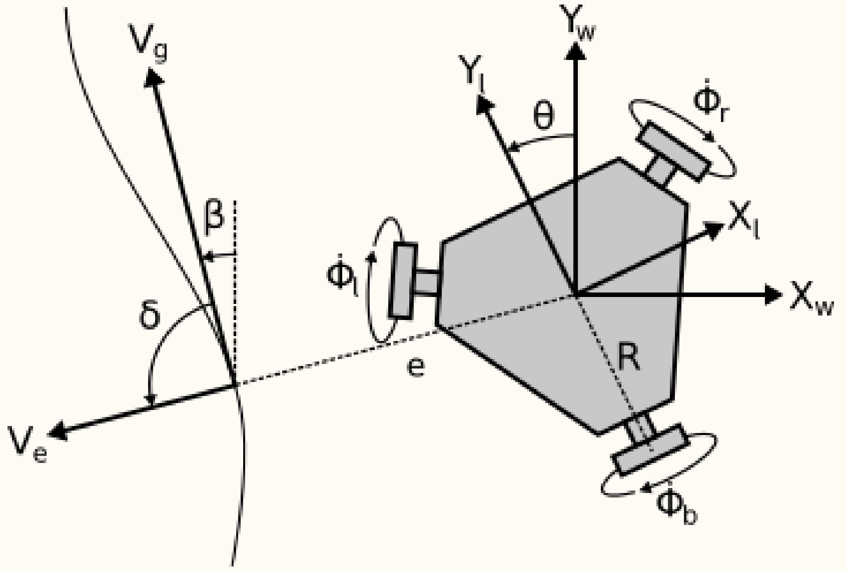
> Differential Drive

> Ackermann Steering

> Omnidirectional Drive







http://dx.doi.org/10.1007/978-3-319-64107-2_47
https://www.quora.com/How-should-I-design-for-Ackermann-geometry
https://www.unmannedlab.org/research/auto-cones

Differential Drive

Characteristics

> Two independently driven wheels and a passive caster for balance.

> Designed for vehicles with nonholonomic constraints, meaning they can't move sideways.

Differential Drive

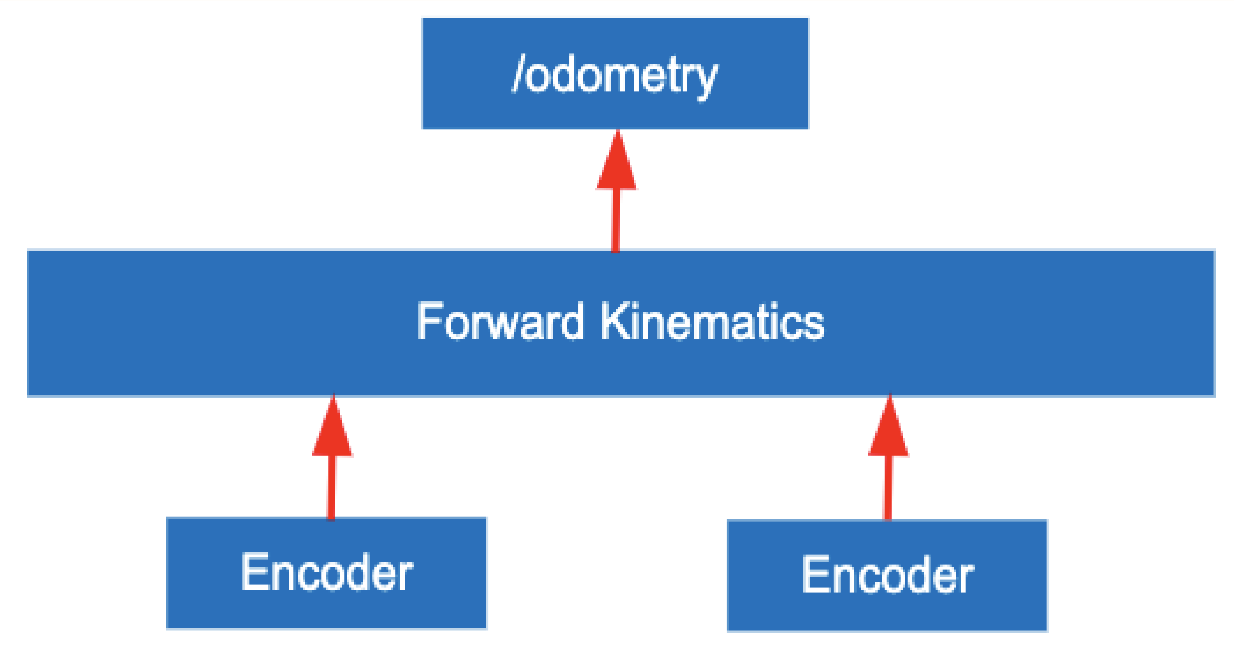
Applications

> Design for simple of development and maintenance. Mostly, it can be found in home use (e.g. Autonomous Delivery Robots, and Robotic Vacuum Cleaners.)

https://irobot.com

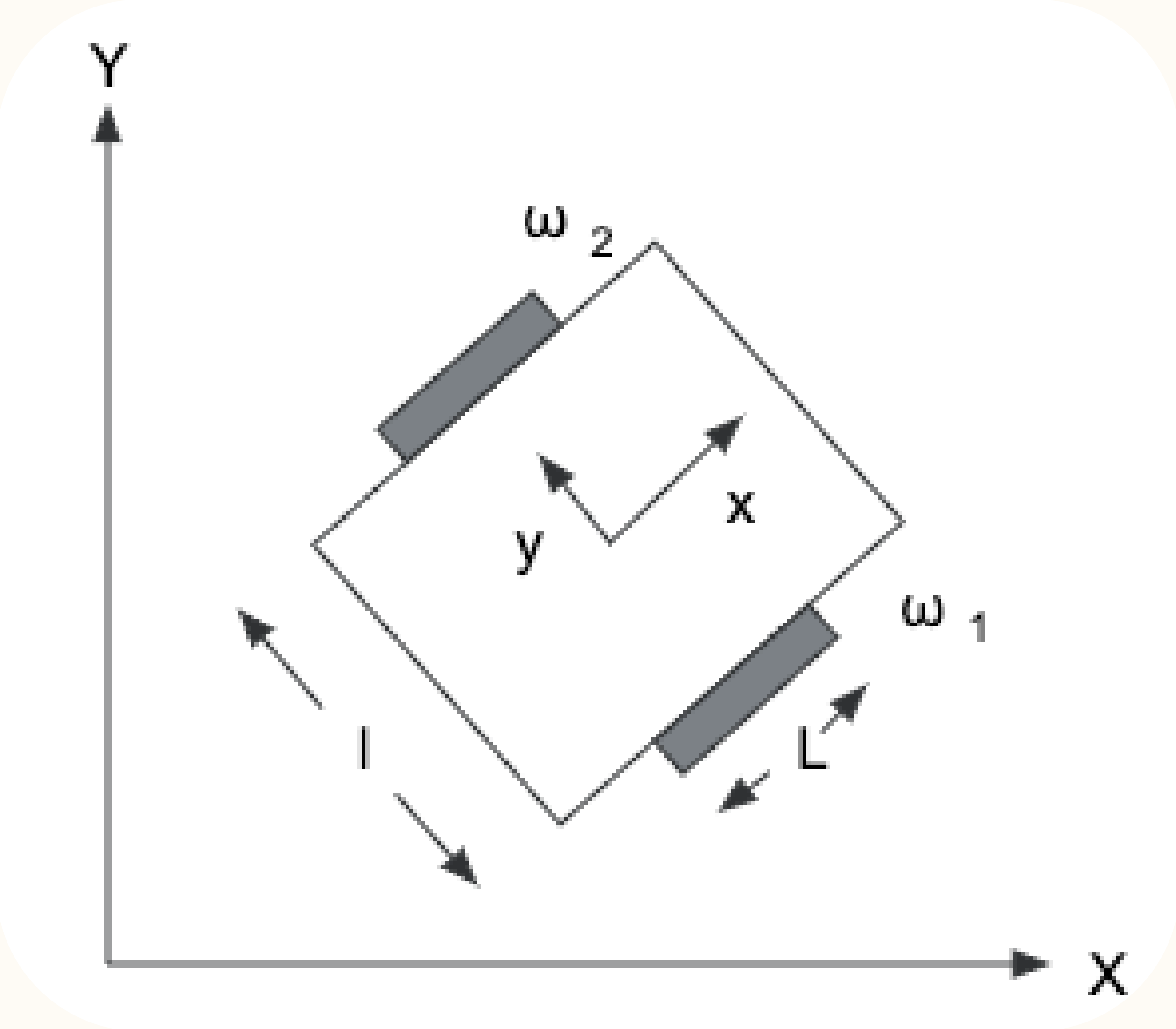
Differential Drive

Forward Kinematics



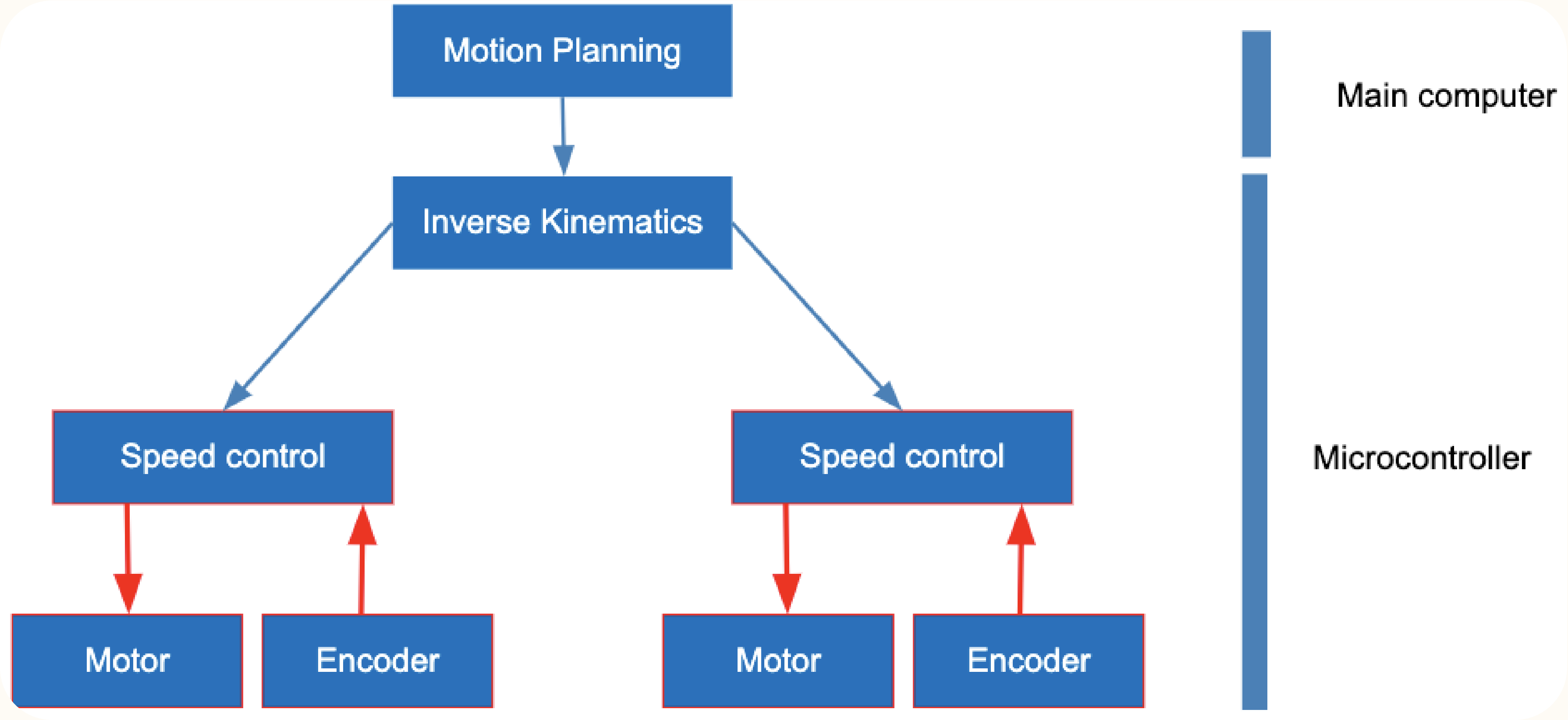
$$v_r = \frac{\omega_1 + \omega_2}{2}$$

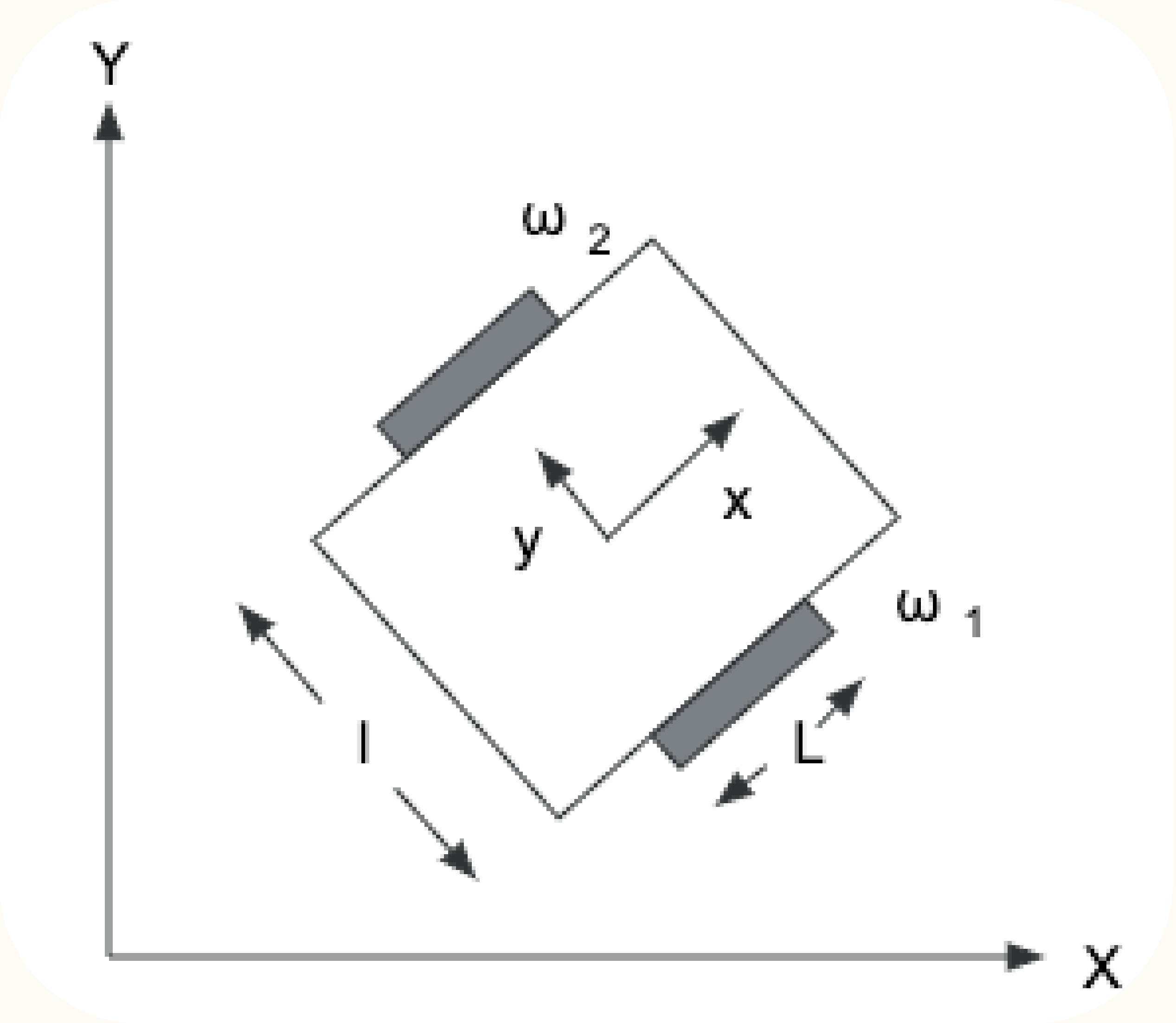
$$\omega_r = \frac{\omega_1 - \omega_2}{l}$$



Differential Drive

Inverse Kinematics





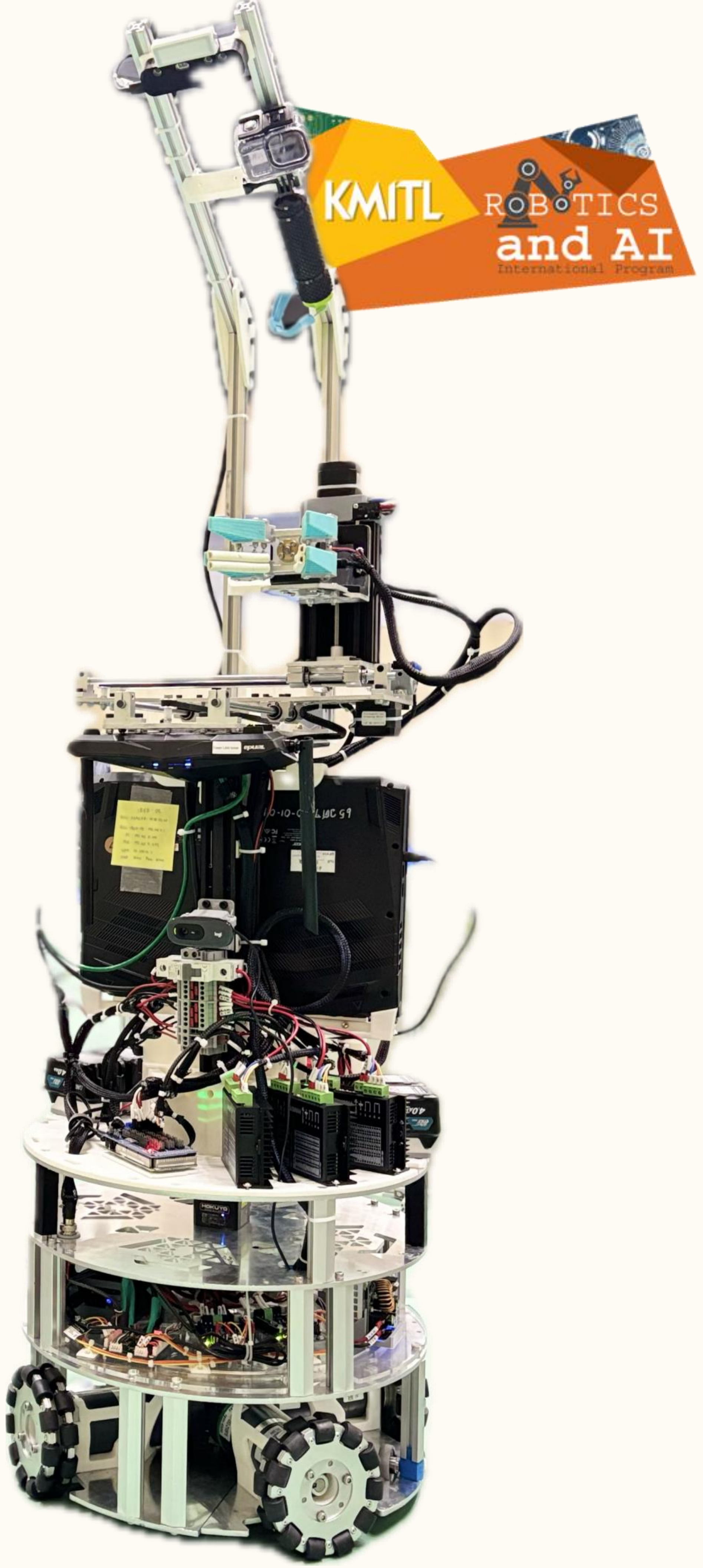
$$\begin{bmatrix}\omega_1\\ \omega_2\end{bmatrix} = \begin{bmatrix}L/2 & L/2\\ L/l & -L/l\end{bmatrix}^T \begin{bmatrix}v_r\\ \omega_r\end{bmatrix}$$

Omnidirectional Drive

Characteristics

> Uses specially designed wheels (e.g., mecanum or omni wheels)

> Exhibits holonomic constraints, meaning it can move in any direction and rotate independently.



Holonomic Systems

Definition

> A holonomic system is one where the number of controllable degrees of freedom (DoF) matches the number of total degrees of freedom the system can physically achieve. (Move in **<u>any direction</u>** in its operational space, without any constraints)

Holonomic Systems

Characteristics

> **Full Mobility** - The robot can translate (move in x, y, and sometimes z directions) and rotate simultaneously.

> **Immediate Response**: The robot can achieve any desired position or orientation without needing intermediate steps or maneuvering.

Nonholonomic Systems

Definition

> A nonholonomic system in robotics is a system where the robot's motion is **constrained**, meaning it has **fewer** controllable degrees of freedom (DoF)

Nonholonomic Systems

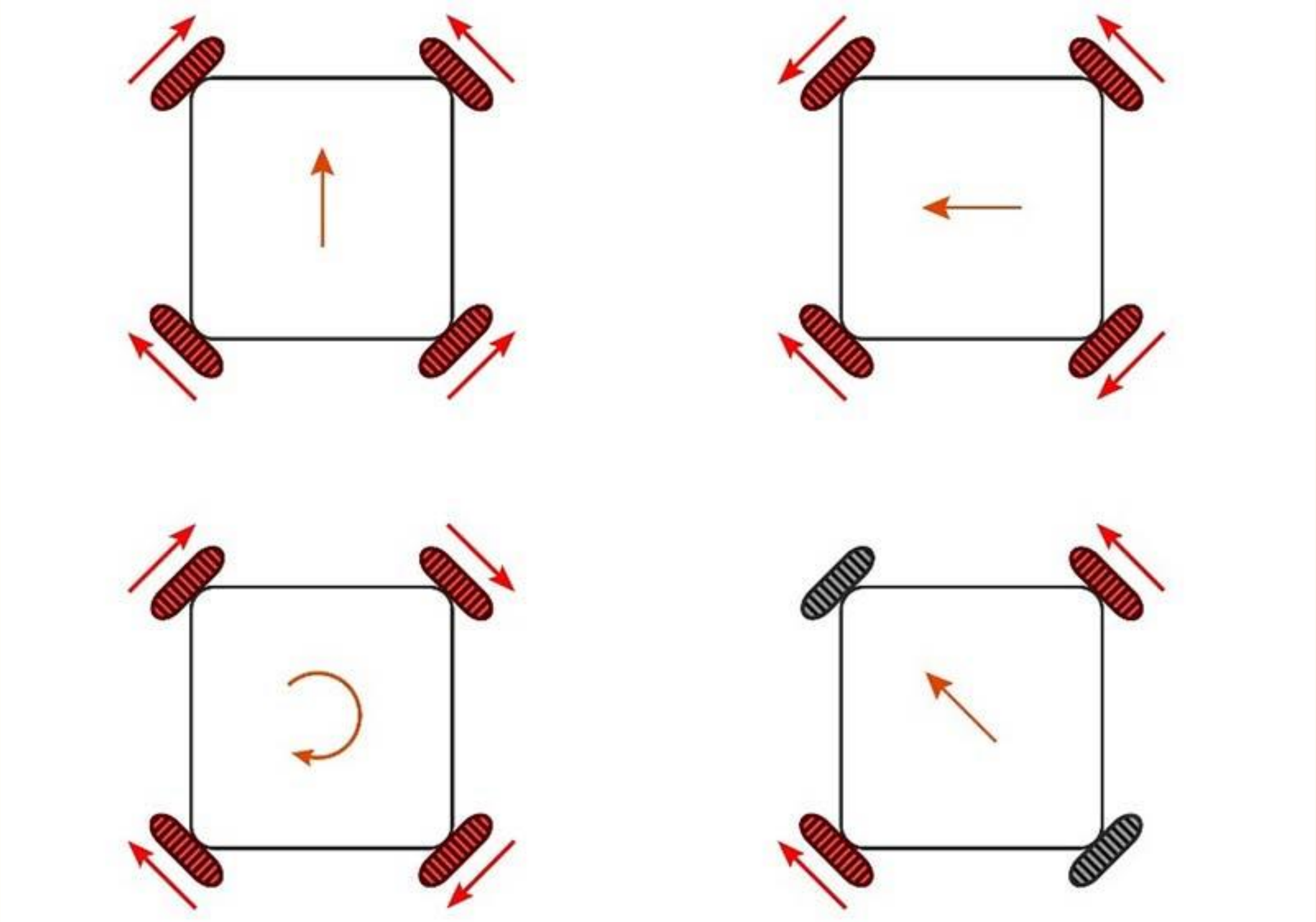
Characteristics

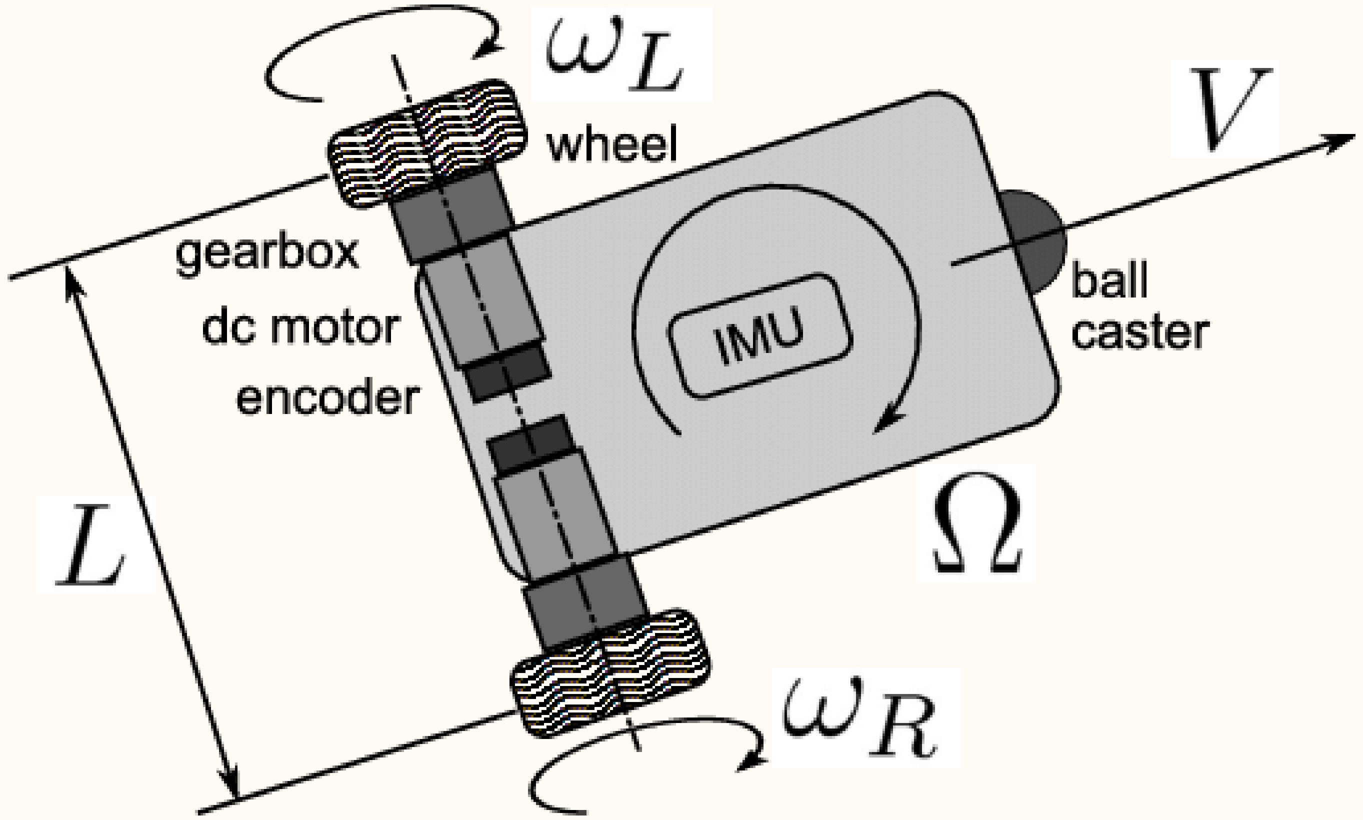
> **Movement Constraints** - The robot cannot directly move sideways or in certain directions; it must reorient or follow a specific path to reach a target position.

> **Path Dependency**: Movement often requires a planned sequence of maneuvers to achieve the desired position and orientation.

Holonomic Systems

vs Nonholonomic Systems



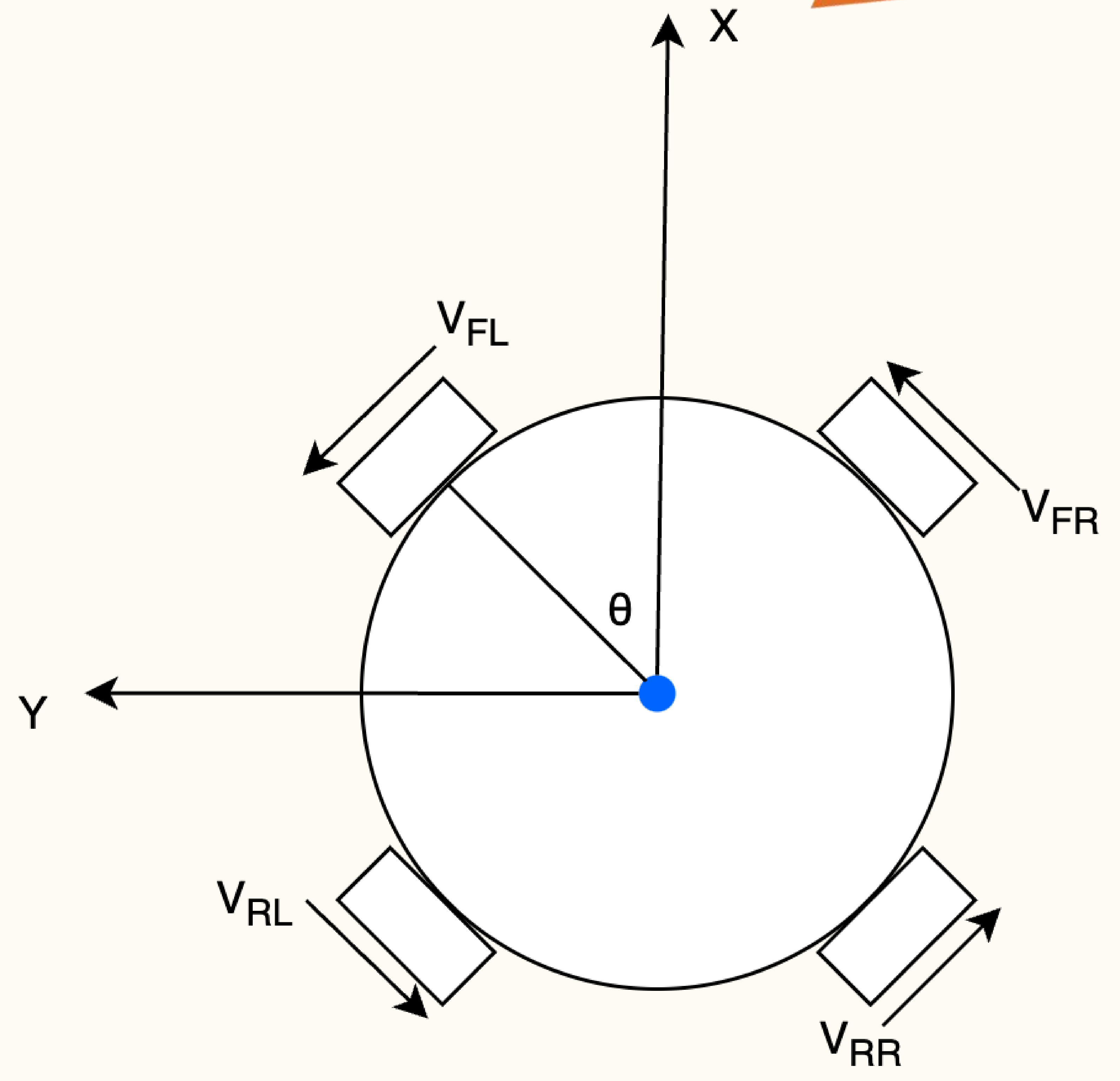


https://www.servomagazine.com/magazine/article/get-rolling-with-omni-directional-wheels
http://dx.doi.org/10.1007/978-3-319-64107-2_47

Omnidirectional Drive

Configuration

> These wheels are mounted at 45-degree angles relative to the robot's frame (V_x)



https://irobot.com

Omnidirectional Drive

Applications

> Ideal for robots that need to navigate tight spaces, such as warehouse or manipulate with machine.

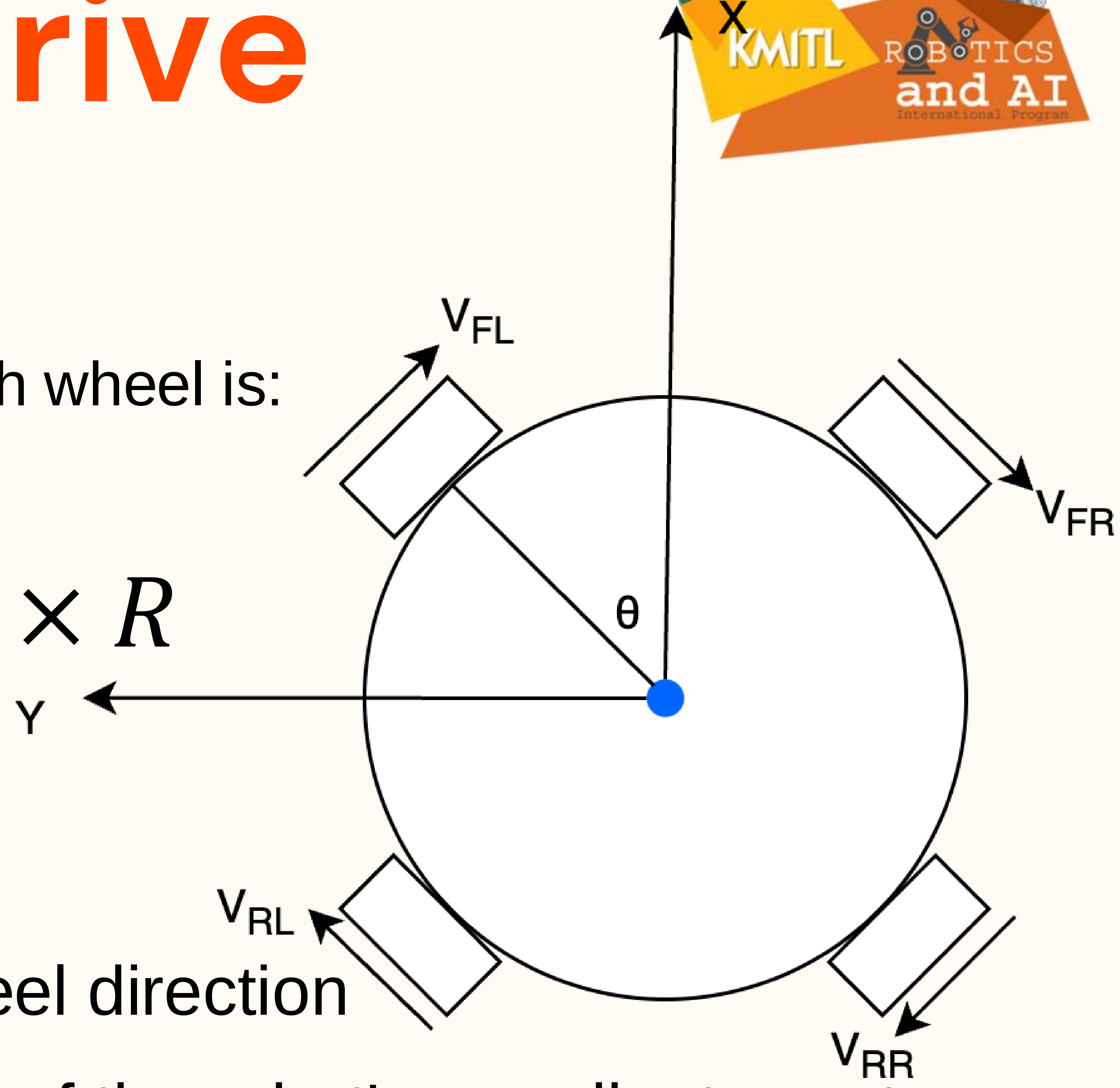
https://irobot.com

Omnidirectional Drive

Inverse kinematics

> The standard form for computing the velocity of each wheel is:

$$v_i = v_x cos(\theta_i) + v_y sin(\theta_i) + \omega \times R$$



** Project linear velocity in x/y-direction onto wheel direction

** θ is measured CCW from the positive x-axis of the robot's coordinate system.

** ω×R represents the contribution of the robot's rotational movement to the velocity of each wheel

Omnidirectional Drive

Inverse kinematics

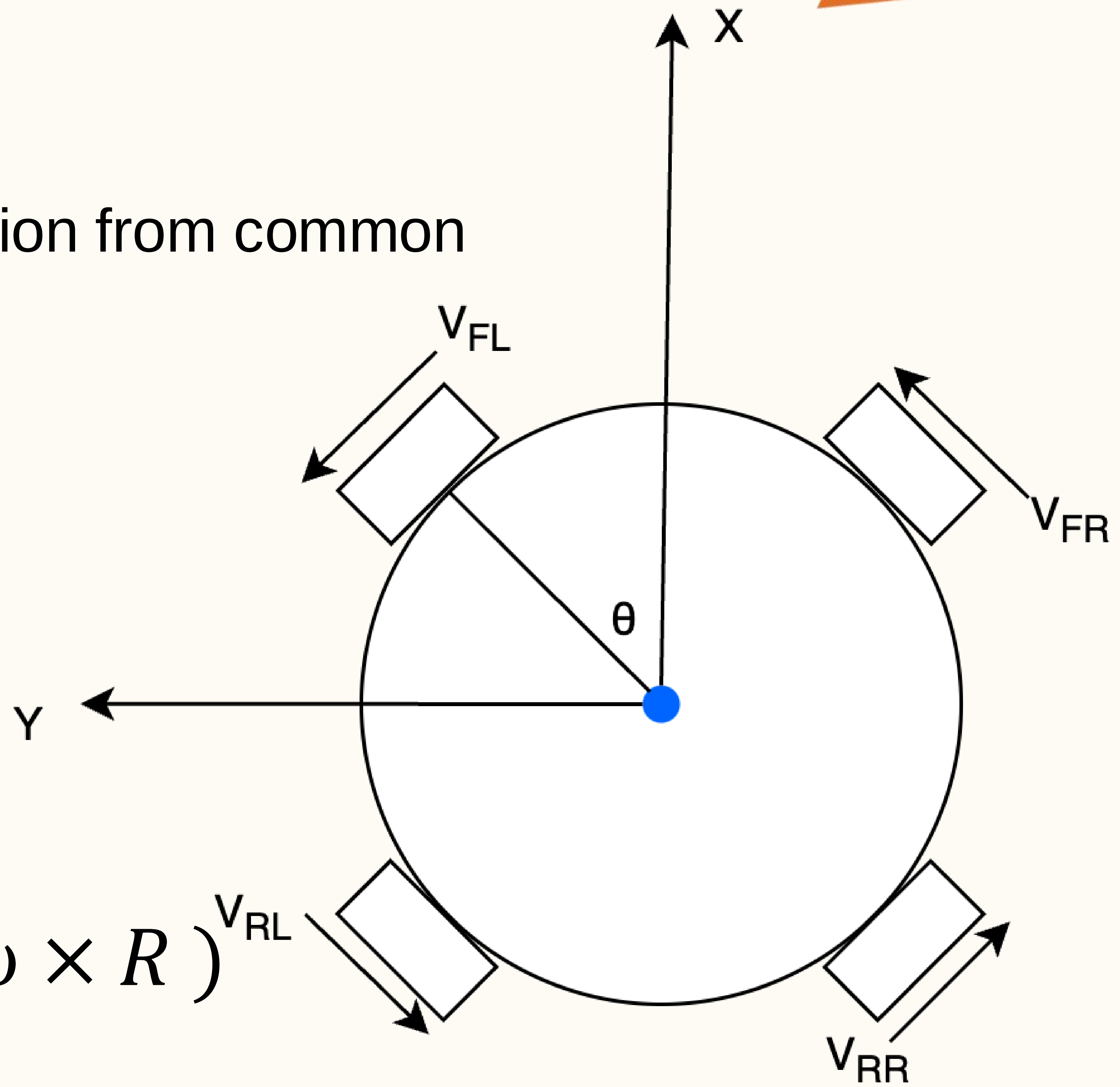
> On practicing, our robot is opposite wheel direction from common

So, we need to

1. Applying a -1 multiplier

2. Swap the roles of cos and sin to ensure the trigonometric functions interact correctly with your wheel configuration.

$$v_i = -1 * (v_x sin(\theta_i) + v_y cos(\theta_i) + \omega \times R)$$



Omnidirectional Drive

Kinematics Implementation

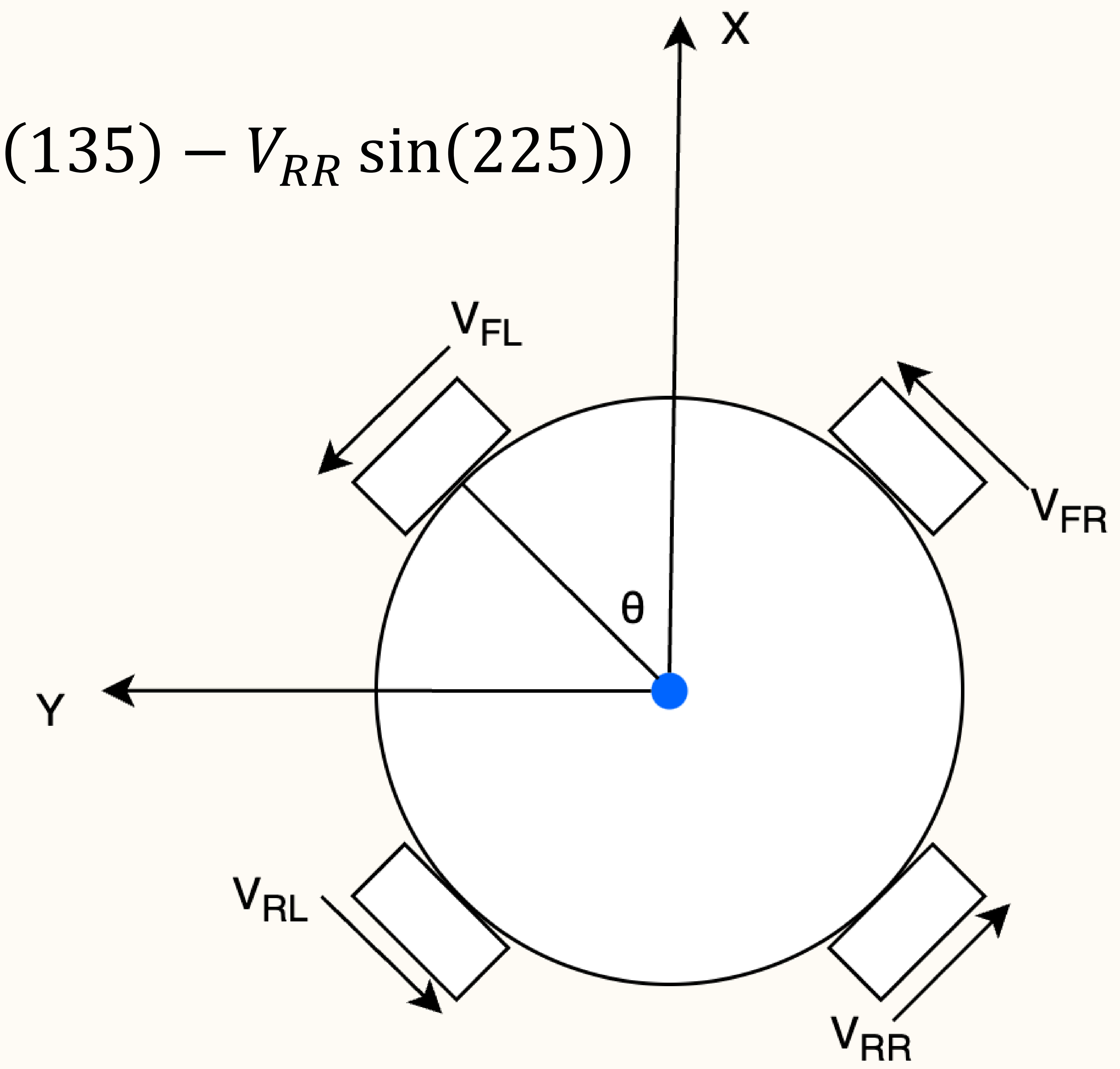
```
def command_velocity_callback(msg):
    last_time_cmd_vel = time()
    goal_linear_x_velocity = msg.linear.x
    goal_linear_y_velocity = msg.linear.y
    goal_angular_velocity = msg.angular.z

    wheel_speed_cmd[FRONT_LEFT] = ?
    wheel_speed_cmd[FRONT_RIGHT] = ?
    wheel_speed_cmd[REAR_LEFT] = ?
    wheel_speed_cmd[REAR_RIGHT] = ?

    set_speed_motor[FRONT_LEFT] = MPS2RPM(wheel_speed_cmd[FRONT_LEFT])
    set_speed_motor[FRONT_RIGHT] = MPS2RPM(wheel_speed_cmd[FRONT_RIGHT])
    set_speed_motor[REAR_LEFT] = MPS2RPM(wheel_speed_cmd[REAR_LEFT])
    set_speed_motor[REAR_RIGHT] = MPS2RPM(wheel_speed_cmd[REAR_RIGHT])
```

Omnidirectional Drive

Forward kinematics

$$v_x = -\frac{1}{2}\left(-V_{FL}\sin(45) - V_{FR}\sin(315) - V_{RL}\sin(135) - V_{RR}\sin(225)\right)$$

** The angles 45, 315, 135, and 225 correspond to the orientations of these wheels relative to the robot's coordinate system.

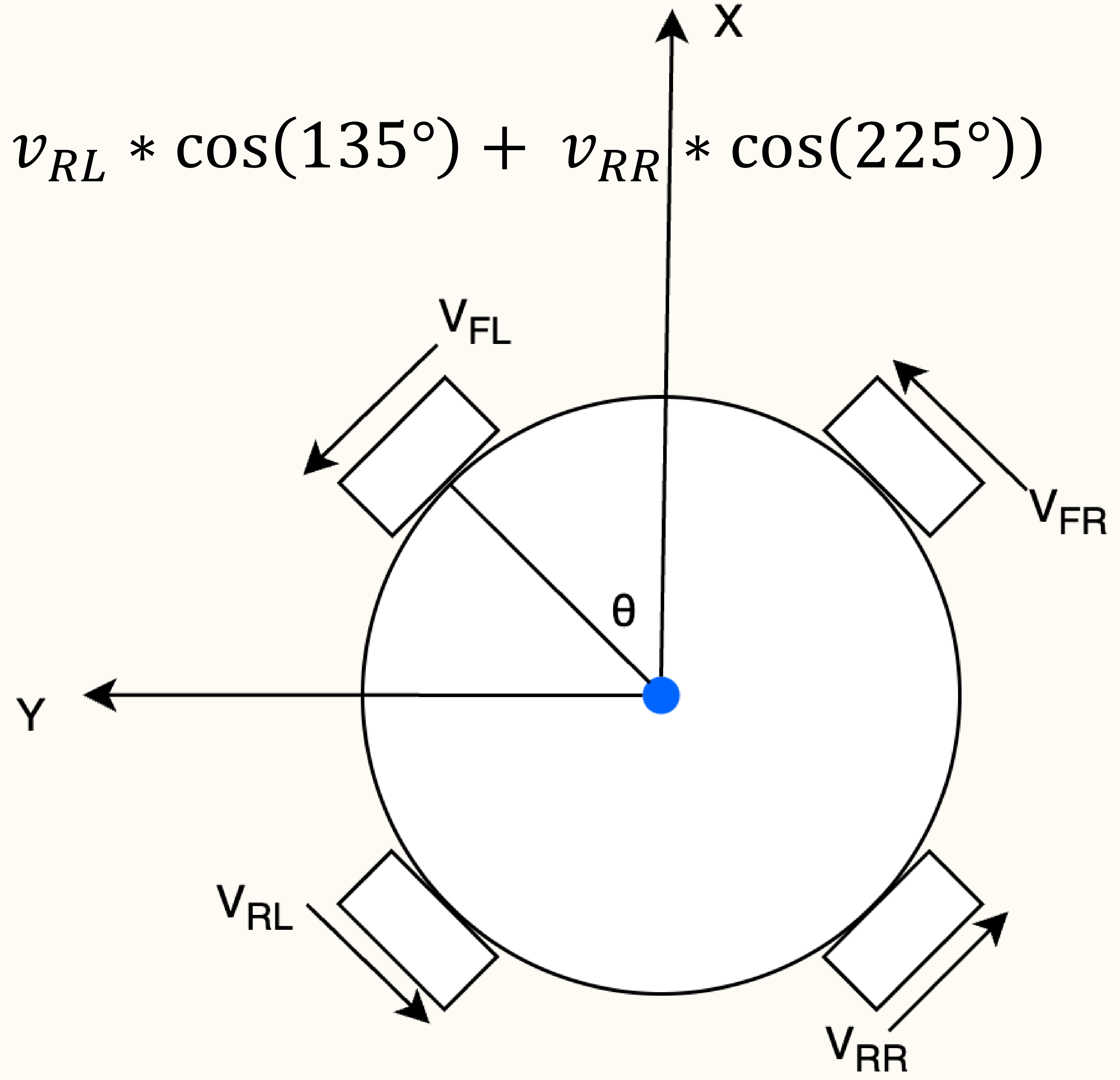


Omnidirectional Drive

Forward kinematics

$$v_y = -\frac{1}{2} * \left(v_{FL} * \cos(45°) + v_{FR} * \cos(315°) + v_{RL} * \cos(135°) + v_{RR} * \cos(225°)\right)$$

** The angles 45, 315, 135, and 225 correspond to the orientations of these wheels relative to the robot's coordinate system.

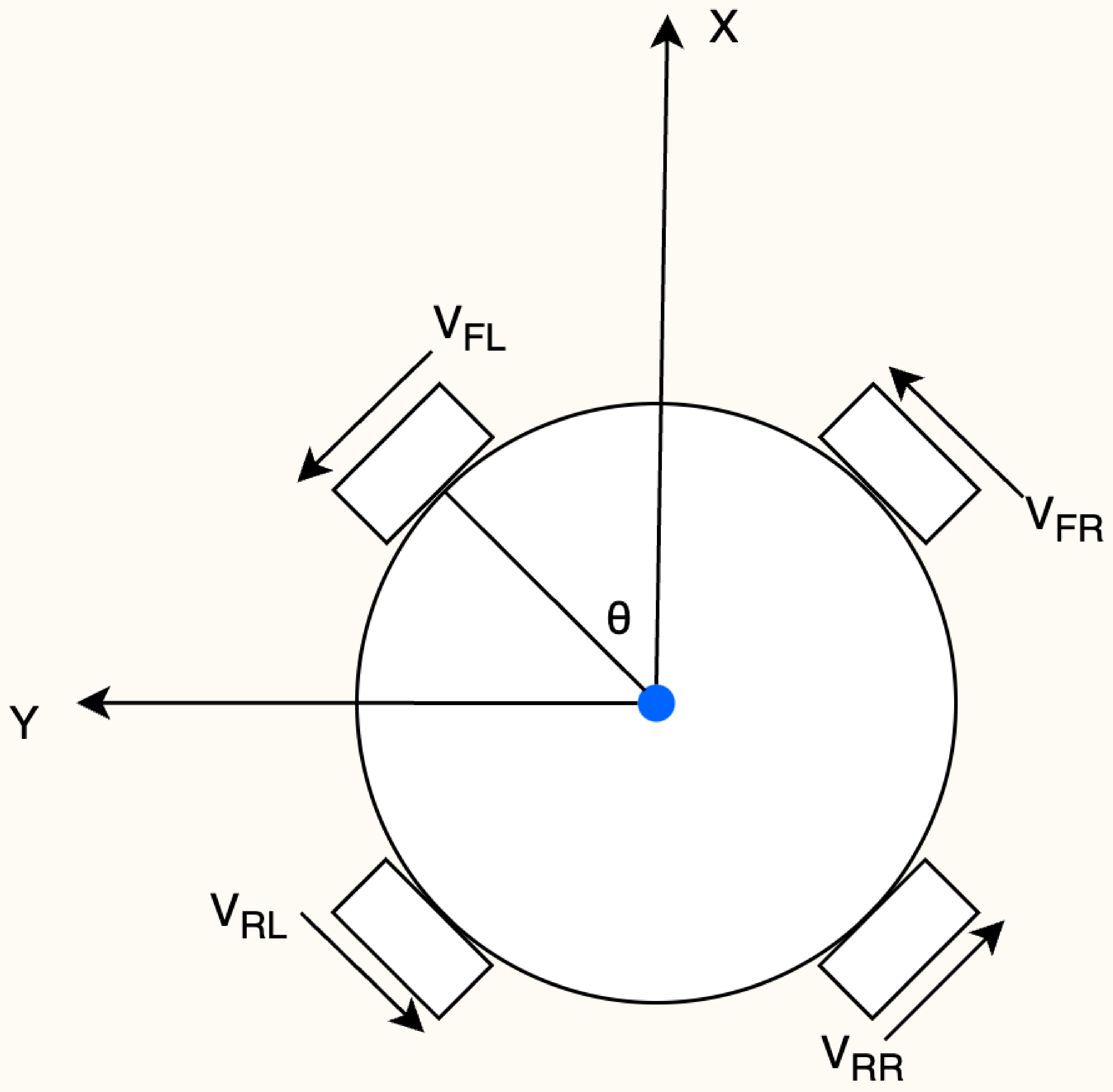


Omnidirectional Drive

Forward kinematics

$$\omega_z = -\frac{1}{4 * R} * (v_{FL} + v_{FR} + v_{RL} + v_{RR})$$

** The angles 45, 315, 135, and 225 correspond to the orientations of these wheels relative to the robot's coordinate system.



Omnidirectional Drive

Kinematics

> Then, write into python code.

Reference

- Rijalusalam, D. U., & Iswanto, I. (2021). Implementation Kinematics Modeling and Odometry of Four Omni Wheel Mobile Robot on The Trajectory Planning and Motion Control Based Microcontroller. *Journal of Robotics and Control (JRC)*, 2(5), 448-455. https://doi.org/10.18196/jrc.25121