

RVO2 Introduction and Comparison

蔡宗諺 Tsung-Yen Tsai

RVO2 - An easy-to-use C++ Library

An implementation of Optimal Reciprocal Collision Avoidance (ORCA) formulation

- A distributed and local collision avoidance algorithm for multi-agent simulation
- Each agent navigates independently without explicit communication with other agents
- Widely used in gaming especially for a larger number of agents scenario such as RTS
- Global planner should be given by external application

ORCA Brief Overview

Overview (From the perspectives of agent A and an agent B exists)

- Input:

size (r_A, r_B), position (p_A, p_B), velocity (v_A, v_B), Max Speed (v^{\max}_A),

preferred velocity (A_{pref}), Time horizon τ

- Output:

v'_A -> a collision-free velocity

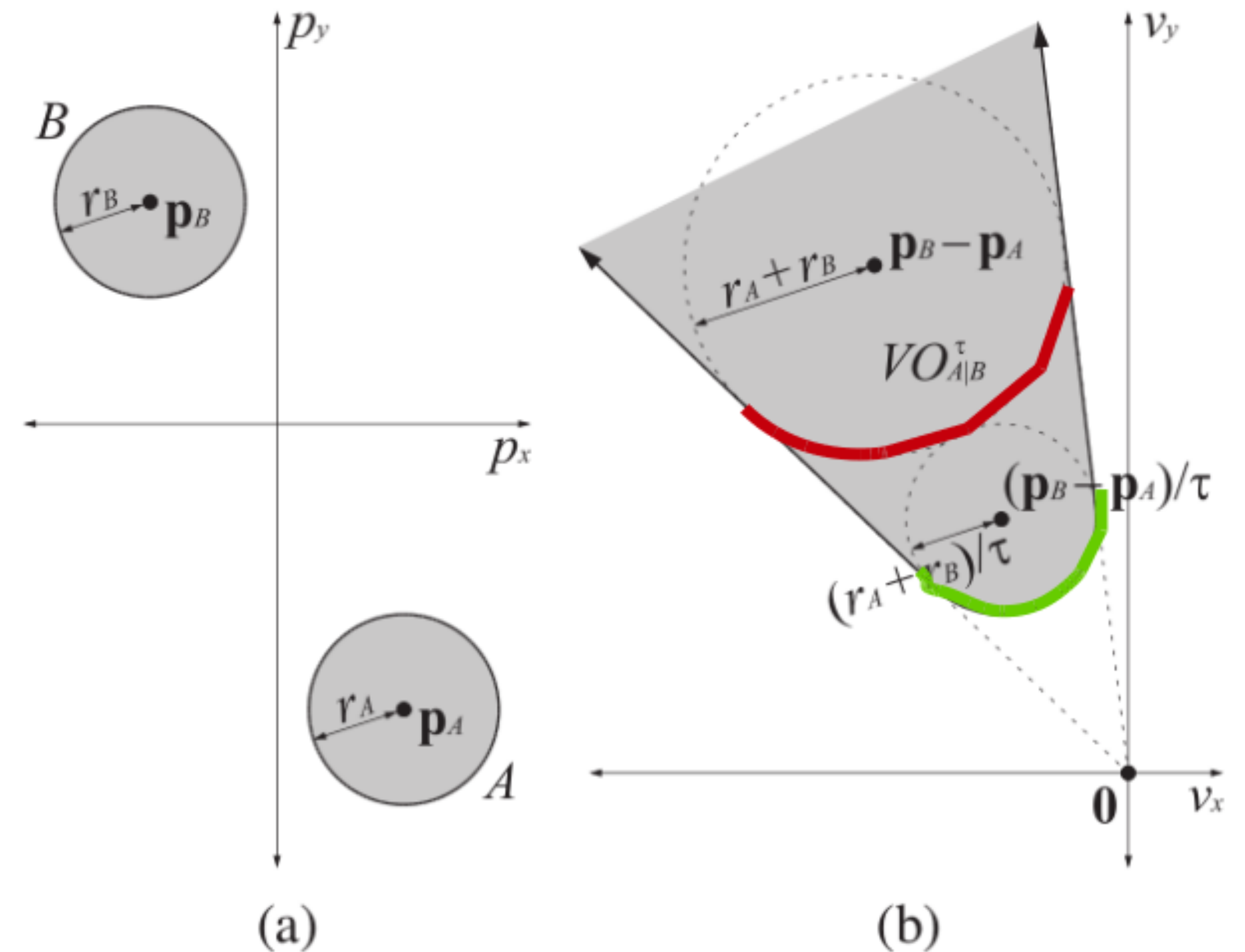
- Feature:

If both agents use ORCA, the resulting v'_A, v'_B are collision-free solution.

(Distributed, without communication)

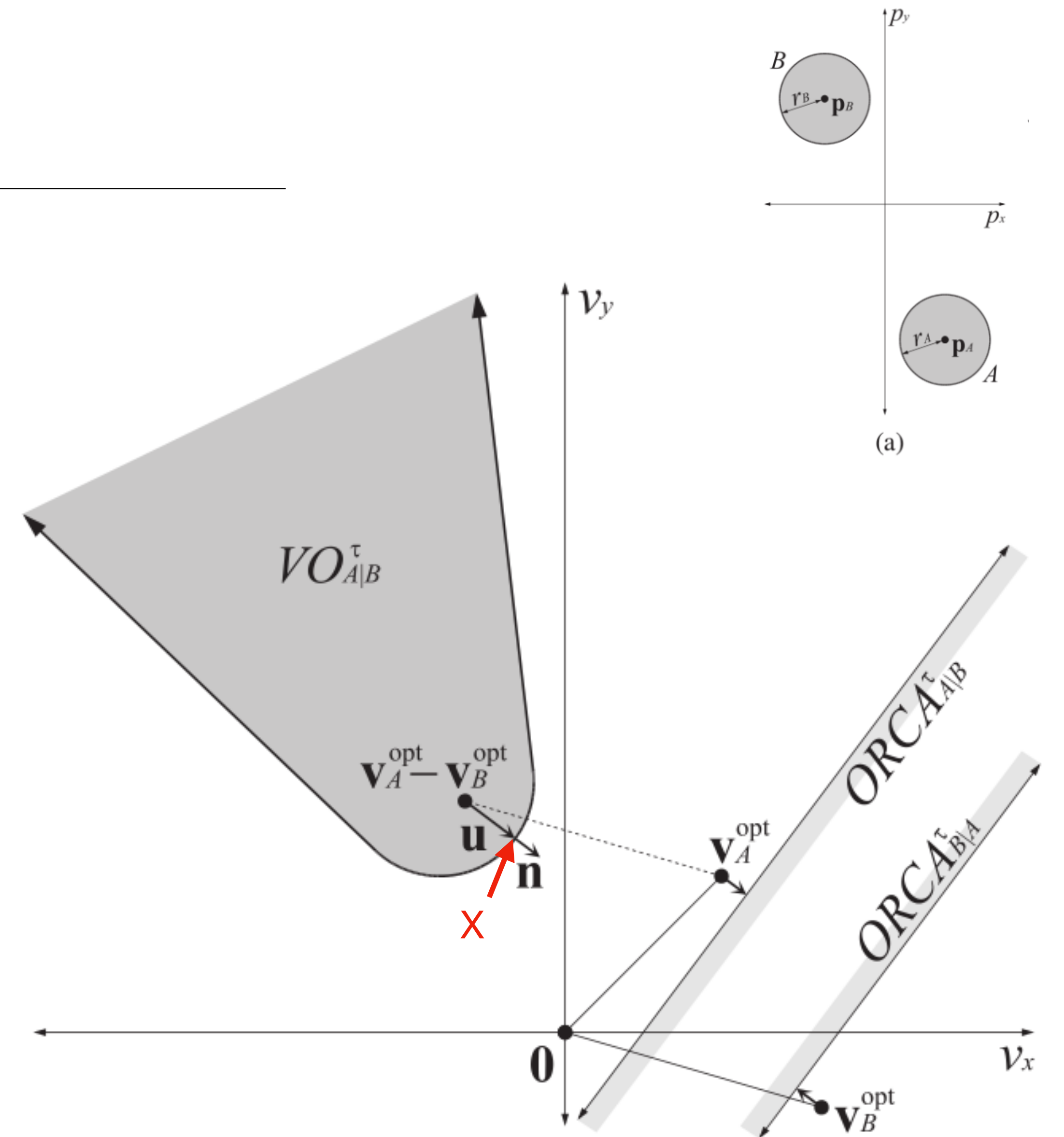
Velocity Obstacle

- Cartesian coordinate
A: r_A, p_A
B: r_B, p_B
- Cartesian coordinate -> Velocity Space:
Take A as a point mass, and assume B is static
Get velocity obstacle -> $VO_{A|B}^\tau$
- If $(v_A - v_B) \in VO_{A|B}^\tau$ -> will collide at some moment before time τ



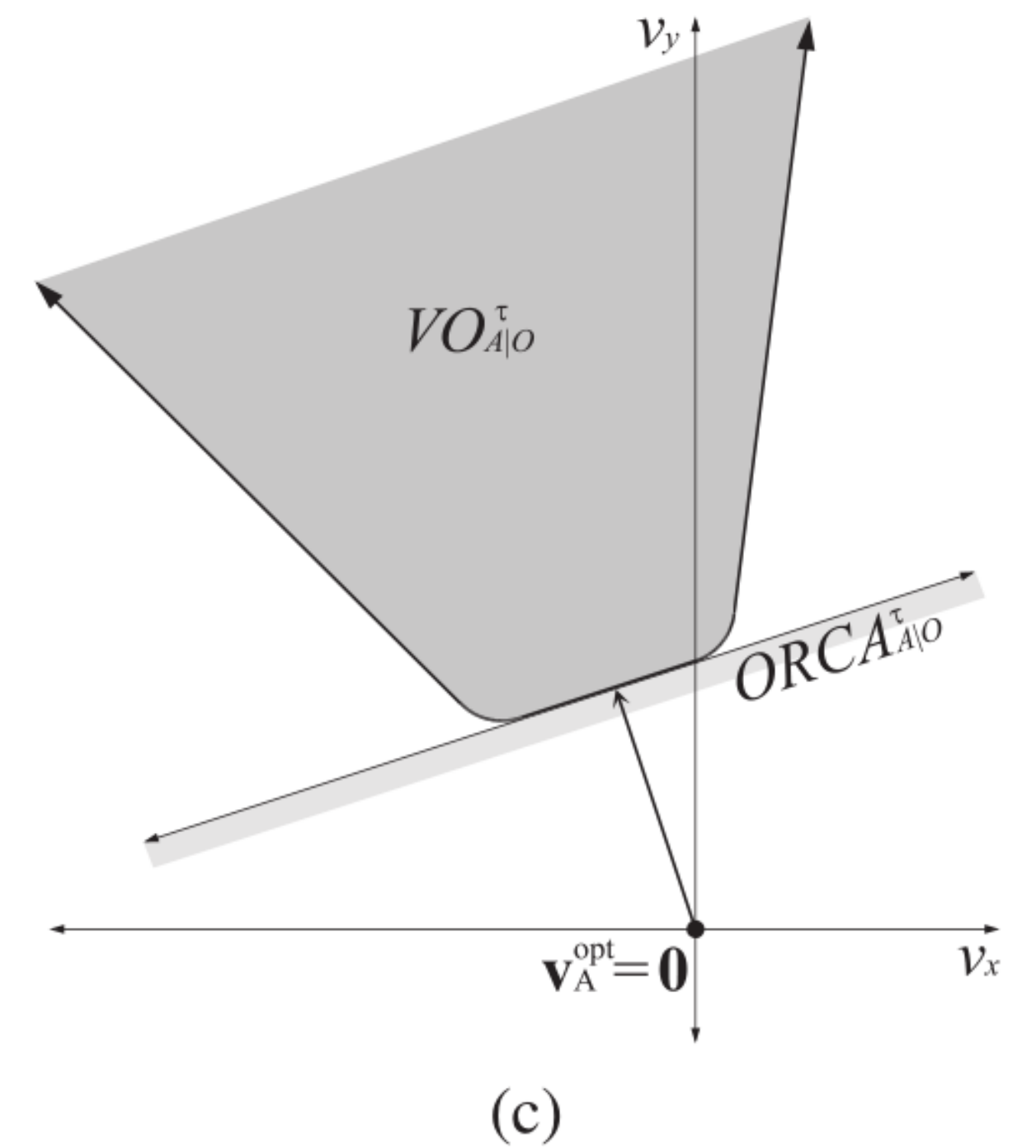
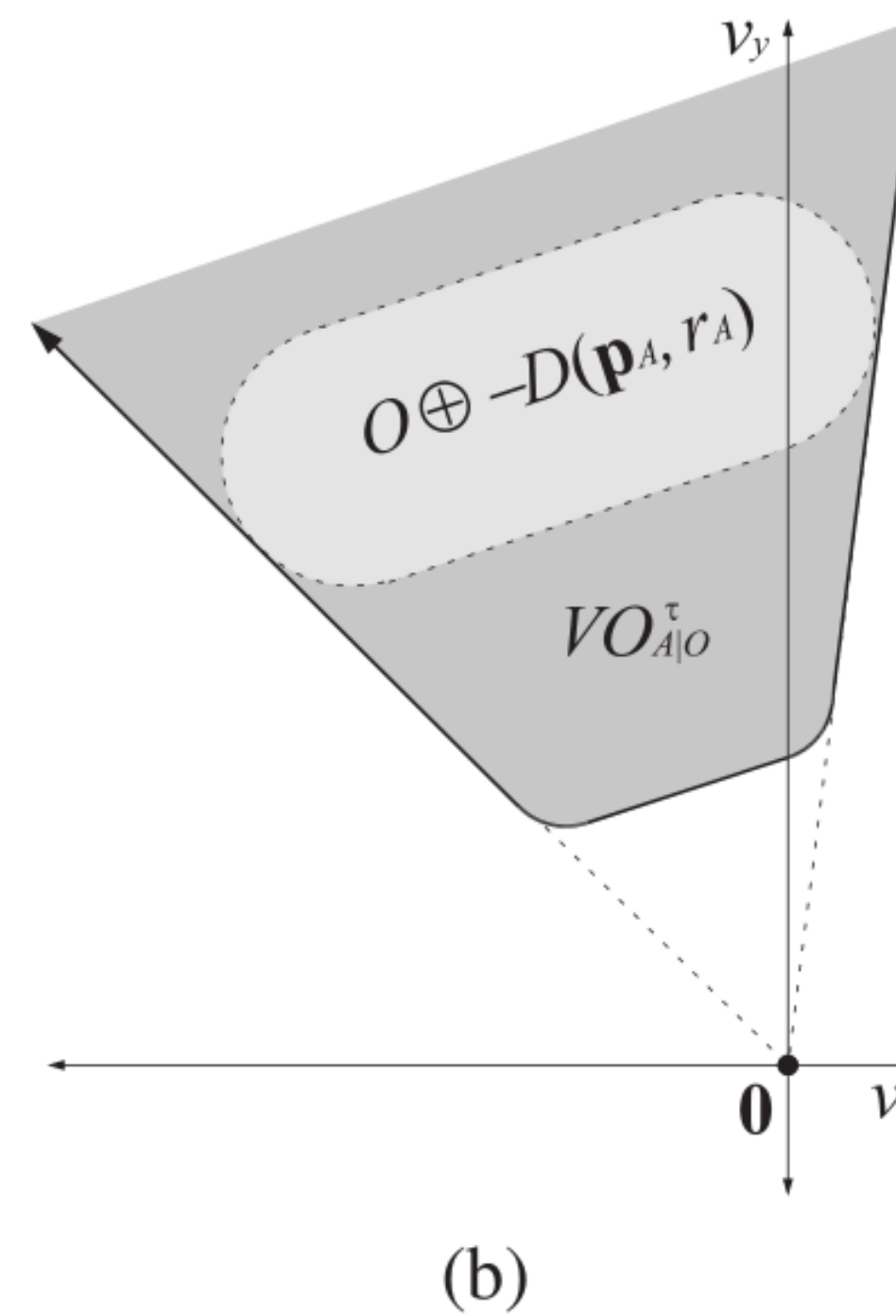
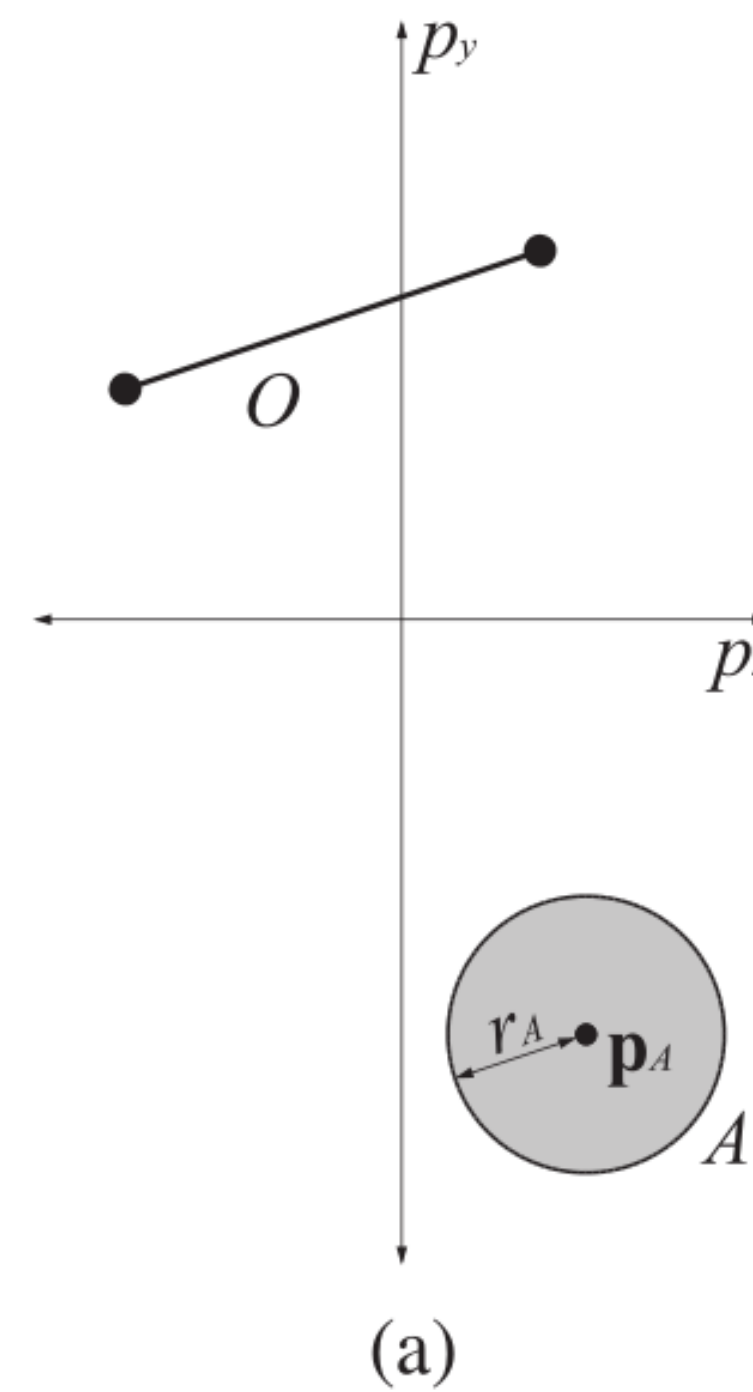
ORCA for dynamic agent

- Find relative velocity $\mathbf{v}_A^{opt} - \mathbf{v}_B^{opt}$
- Find the nearest point \mathbf{x} on $VO_{A|B}^\tau$ boundary and the vector \mathbf{u}
- $\mathbf{v}_A^{opt} - \mathbf{v}_B^{opt} + \mathbf{u} = (\mathbf{v}_A^{opt} + \frac{1}{2}\mathbf{u}) - (\mathbf{v}_B^{opt} - \frac{1}{2}\mathbf{u})$
- The set $ORCA_{A|B}^\tau$ of permitted velocities for A is the half-plane pointing in the direction of \mathbf{n} starting at the point $\mathbf{v}_A^{opt} + \frac{1}{2}\mathbf{u}$



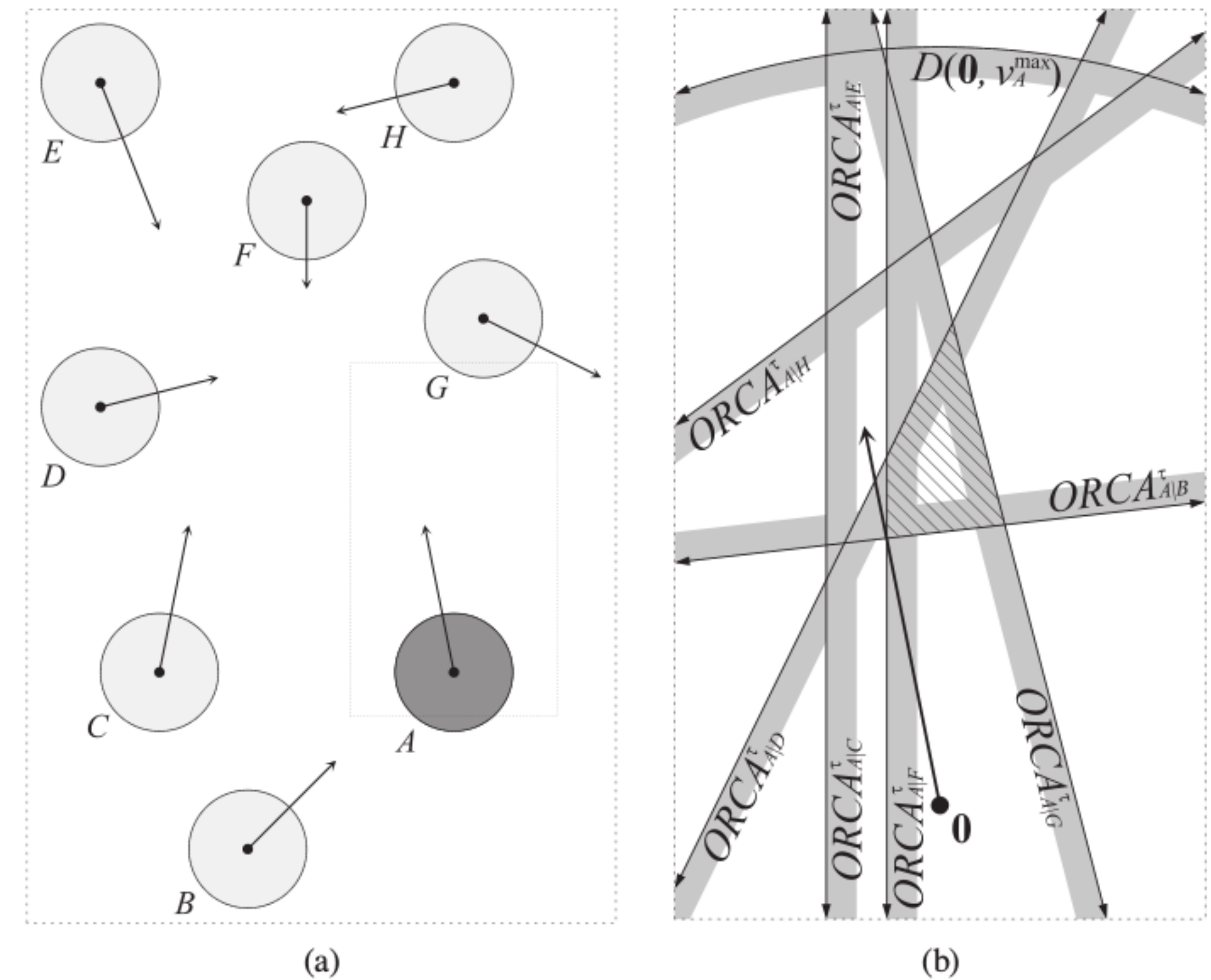
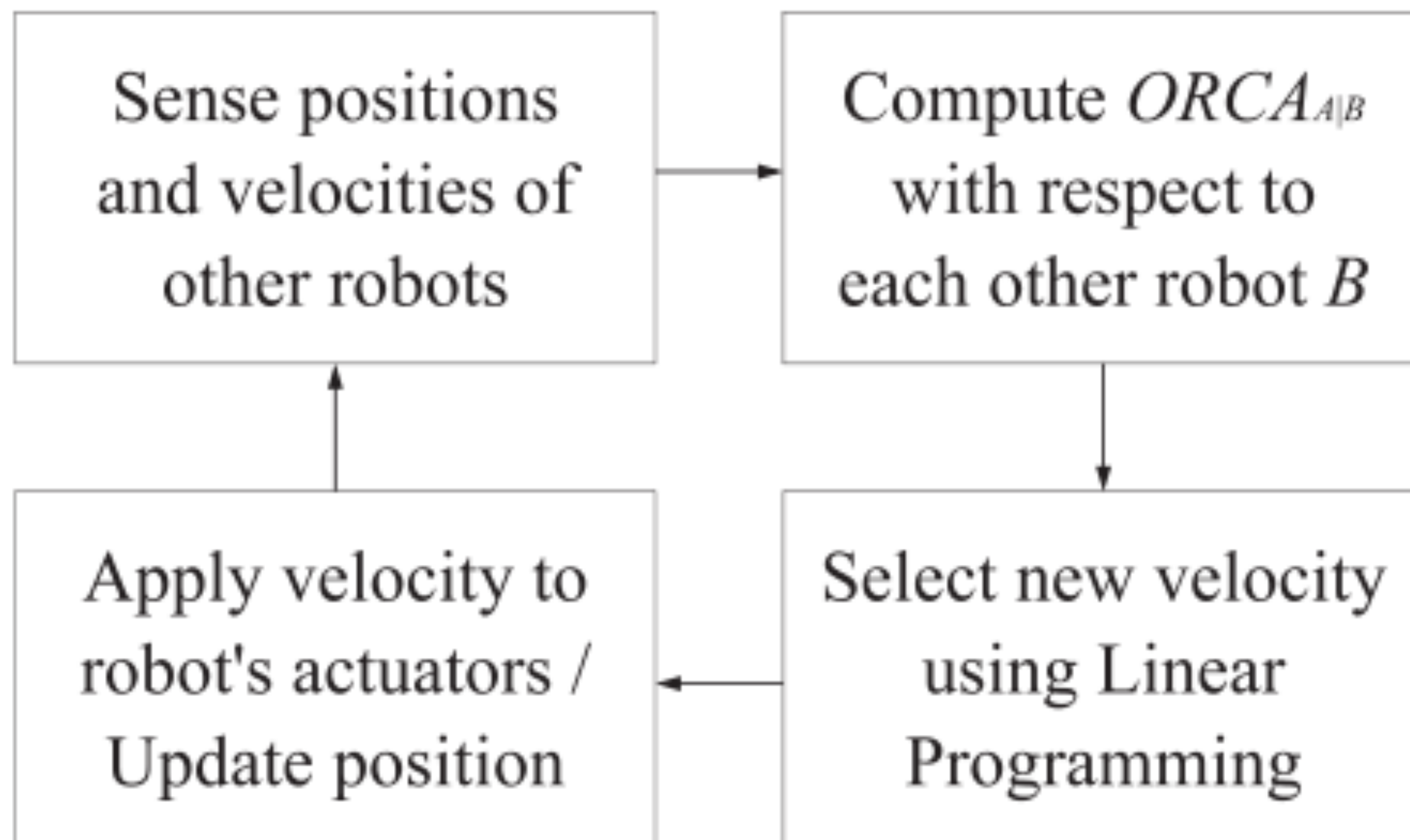
ORCA for static obstacle

- Robot should take full responsibility of avoiding collisions with them.
- $\mathbf{v}_A^{opt} = \mathbf{0}$
- Should take smaller τ



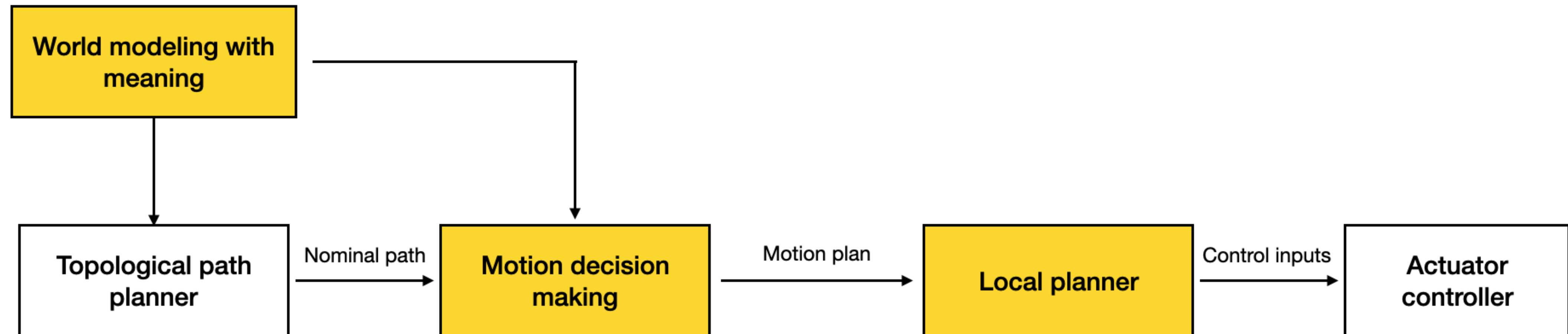
<https://blog.csdn.net/u012740992>

Process of ORCA



- Time complexity $O(n)$, n is the number of robots

Master Research – Approach

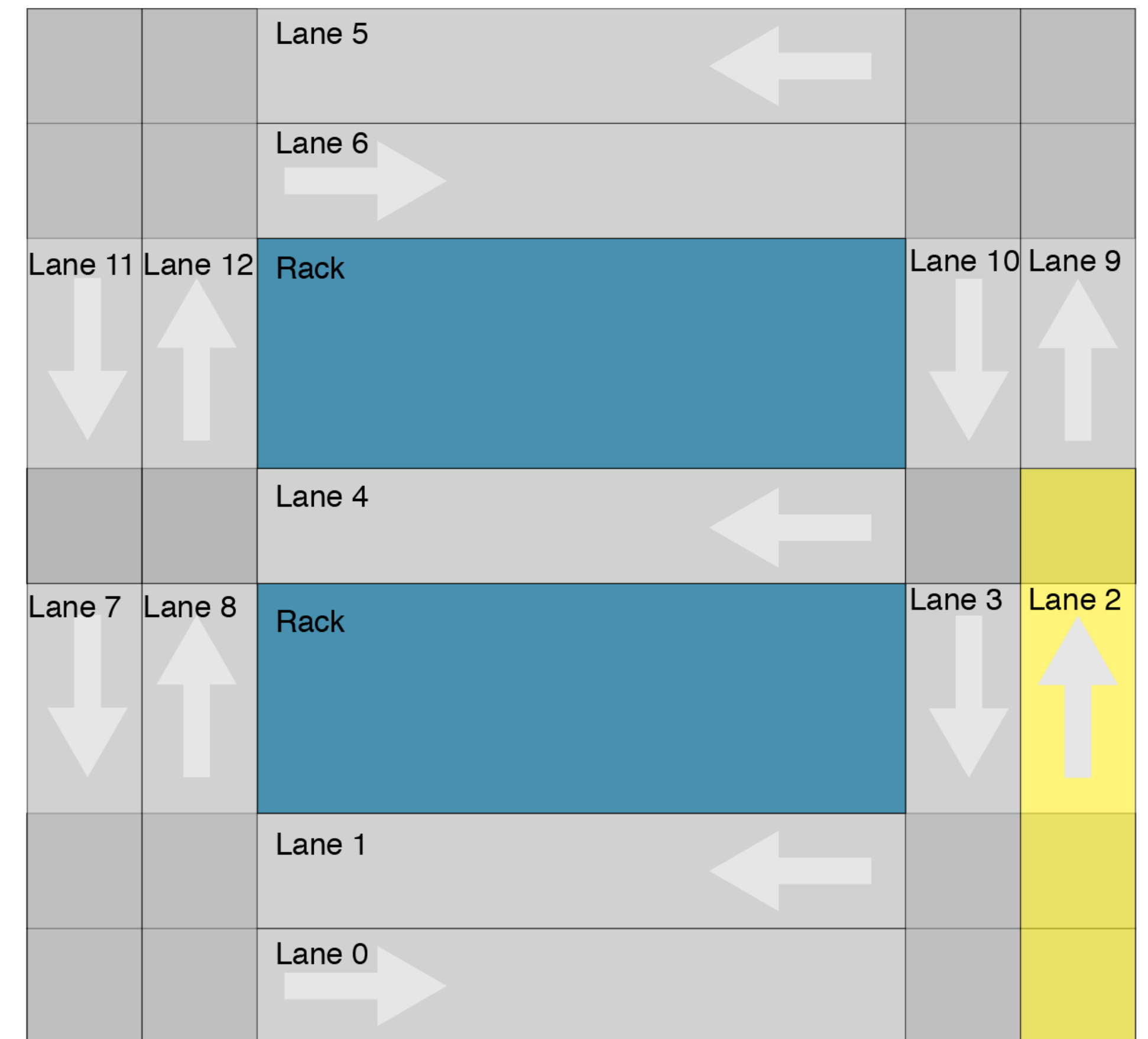


A Distributed Approach: Local motion planner with increased flexibility

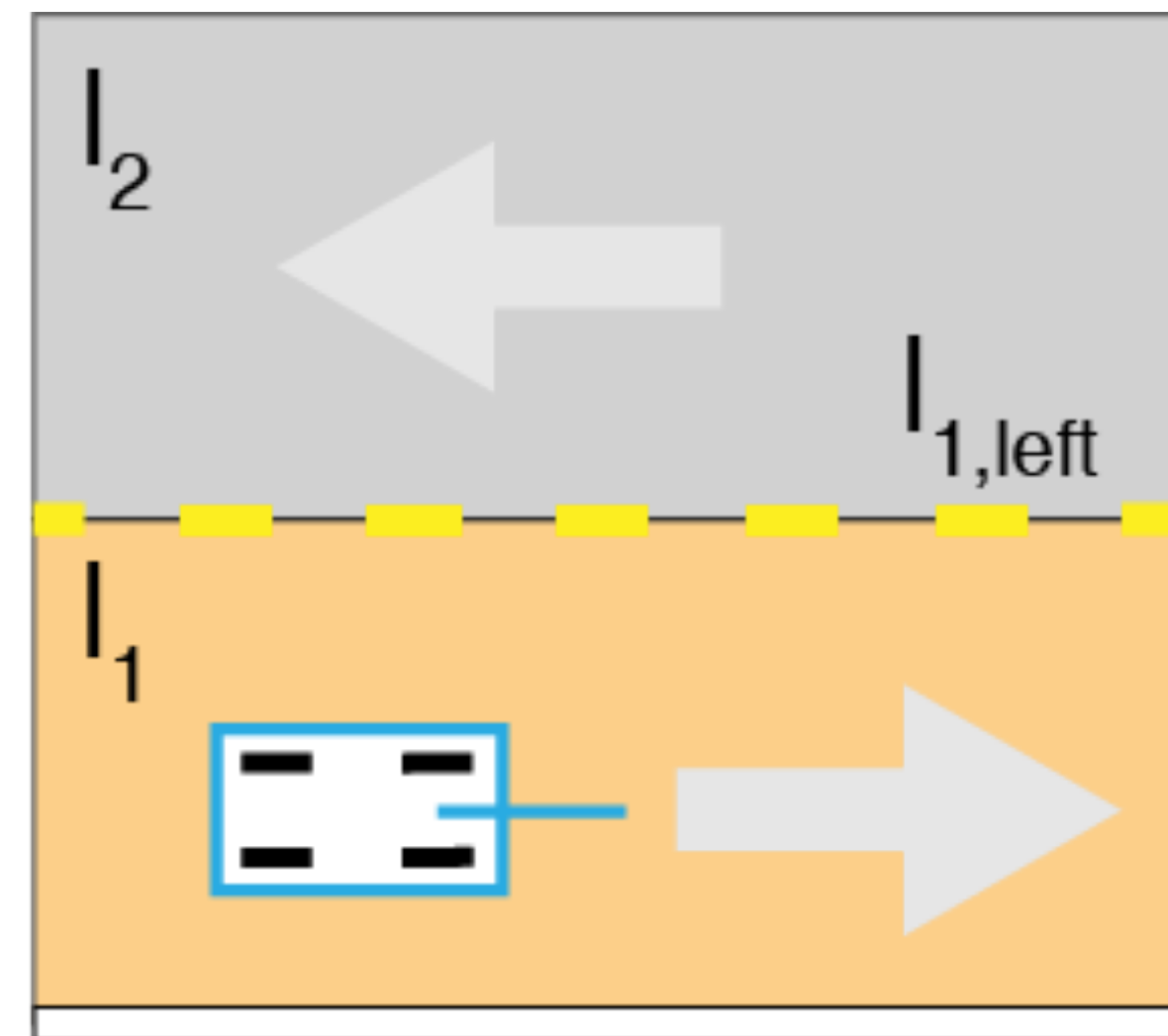
Master Research – Semantic World Model

Semantic world model




- Provide human's prior knowledge on the considered environment
- Virtual lanes system
- Desired travel behavior
- Indication but not constrain



Master Research –Situational aware motion planner



Desired path

Type	Label
 Driving lane	Drivable
 Separation line	Avoid
 Boundary line	No-go

1. Describe the relationship between the robot and relevant objects to be true/false statement (condition)
 2. Using composable conditions to represent specific situation
 3. Generate a reaction plan to cope with the situation
- Situation: The robot drives along the desired path

A. Semantic driving command:

Acceleration = [speedup/maintain], Steering = [ahead]

B. Semantic annotations:

l_1 : drivable, $l_{1,left}$: avoid, $l_{1,right}$: no-go

C. Short prediction horizon:

Default: 1 second

Master Research – Action-based Model Predictive Control

Find the first minimum feasible performance

- > Simplify the cost function to be possible collision exist or not
- > Search space generation with specific direction
- > The starting sample is important

A. Semantic driving command:

Acceleration = [maintain],

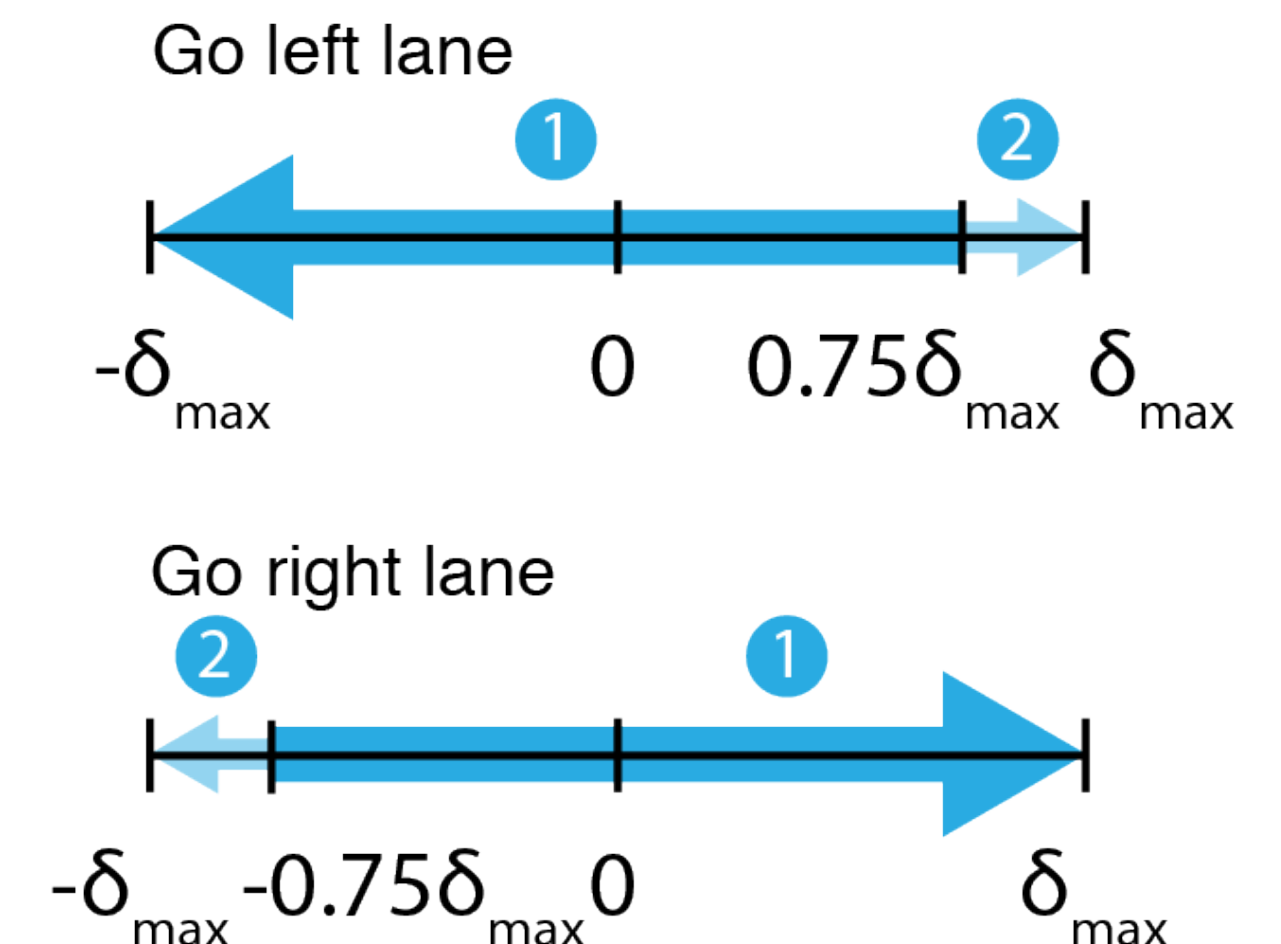
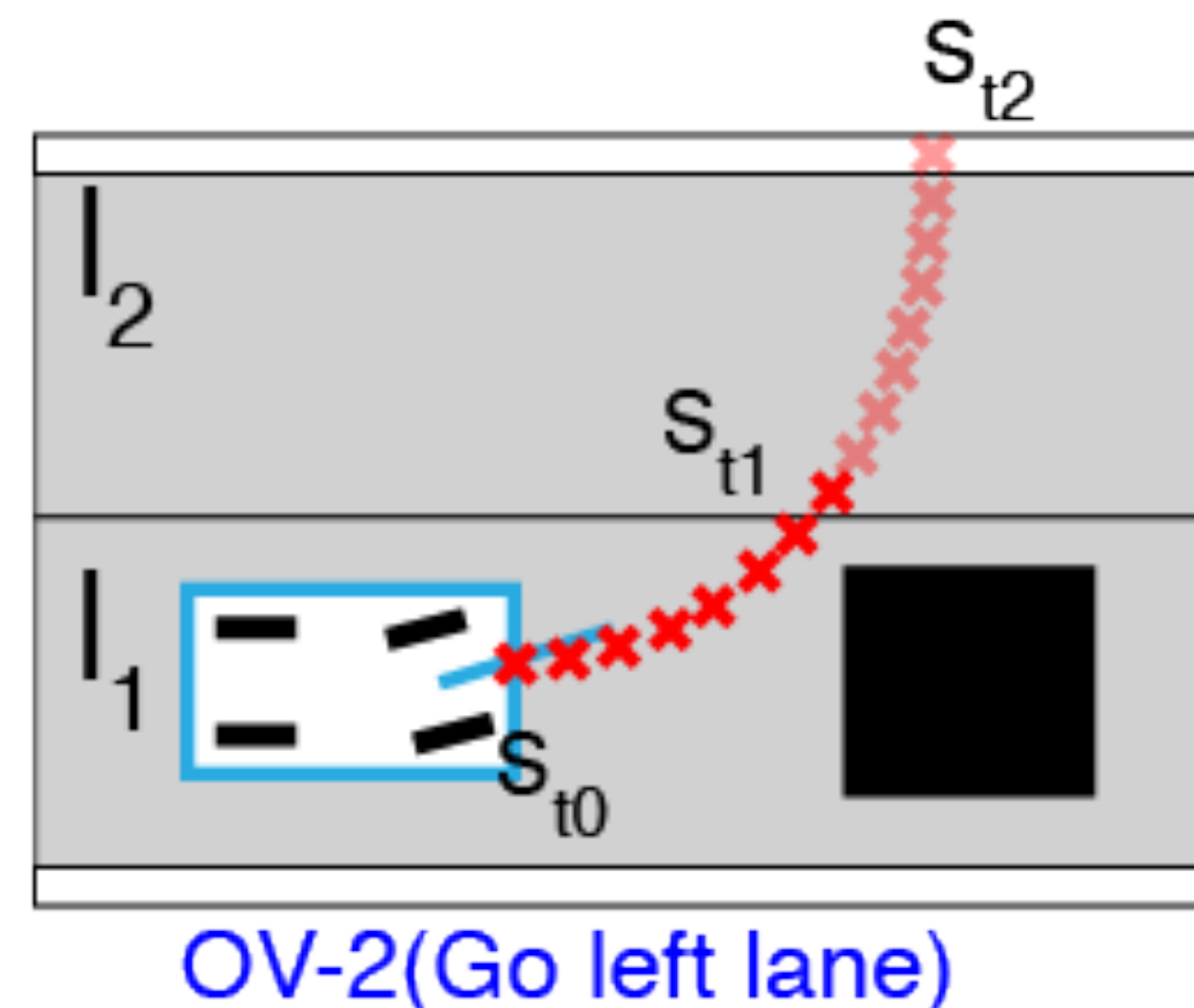
Steering = [Go left lane]

B. Semantic annotations:

l_1, l_2 : drivable, $l_{1,\text{right}}, l_{2,\text{right}}$: no-go

C. Prediction horizon:

2 second

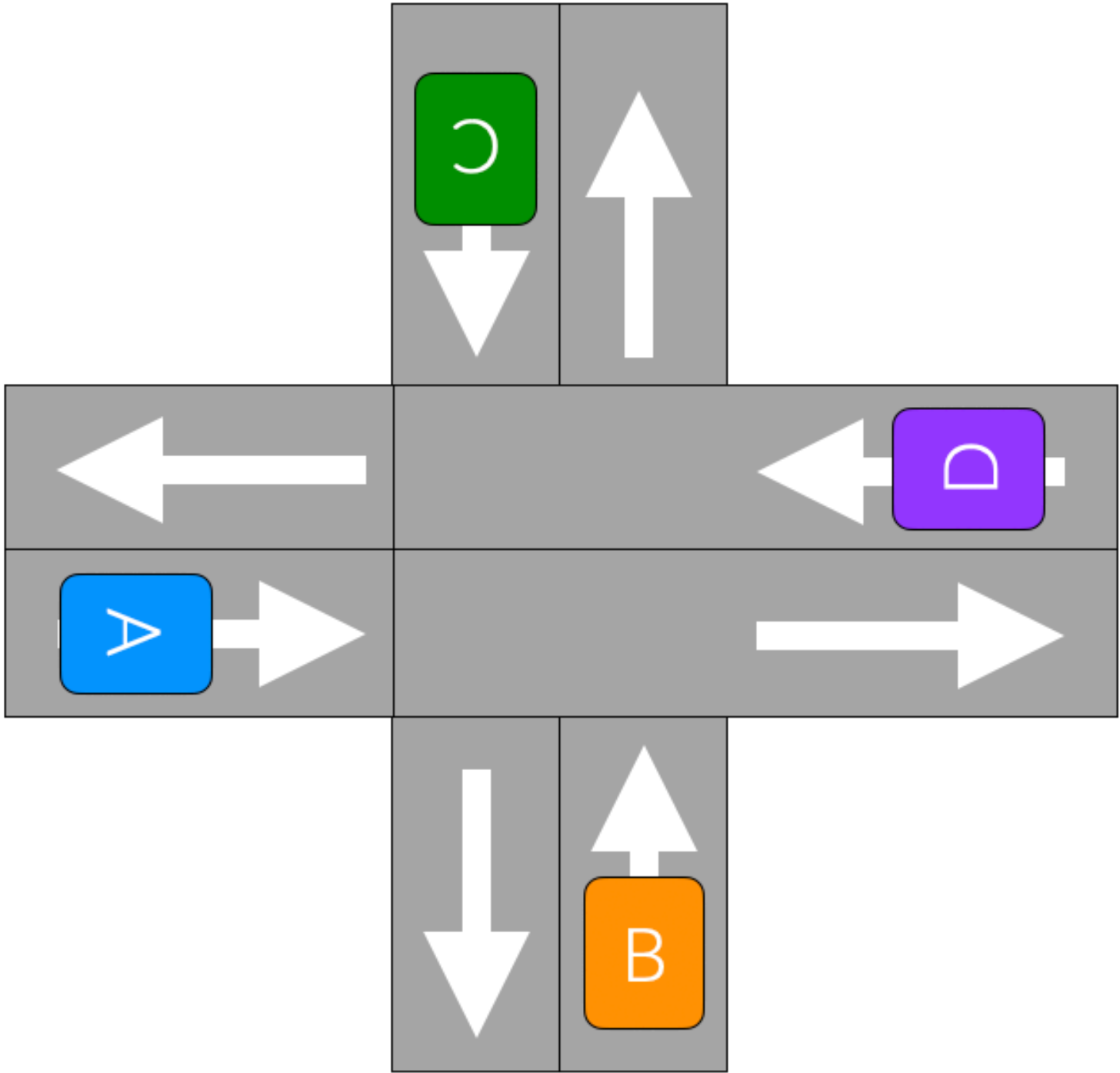


Master Research – Multi-robots coordination

Official traffic regulations

- **Article 15.1** At road junctions, drivers must give priority to traffic approaching from the right.
- **Article 18.2** Drivers intending to turn left must give way to oncoming drivers intending to turn right at the same road junction.

Master Research –Dynamic Perception horizon



Robot's action	Perception horizon for coordination
Turn right	□
Go forward	Right side lane
Turn left	Right side lane and Opposite side lane

Master Research – Conclusion

Multi-robot coordination

1. The robot takes a dynamic perception horizon for coordination
2. It has to pay attention to the traffic from certain direction based on its intersection intention

Drawback

1. The robot has to obtain the reliable and latest information about relevant agents

Comparison between ORCA and SAMP

	ORCA	SAMP
Usage	General	Together with semantic world model and focus on passing an intersection
Centralized/ De-Centralized	Full distributed	Partial distributed
Other agent's info	(Observed) position, size, velocity	(Observed/Shared Map) position, size, velocity, intention
Environment	Open space *1	Structured space
Number of agents	Good at work with a large number of agents	Only coordinate with agents around the intersection
Feature	Able to navigate through a crowed area	Able to perform more sophisticated behavior for certain situation *2

Comparison between ORCA and SAMP *1

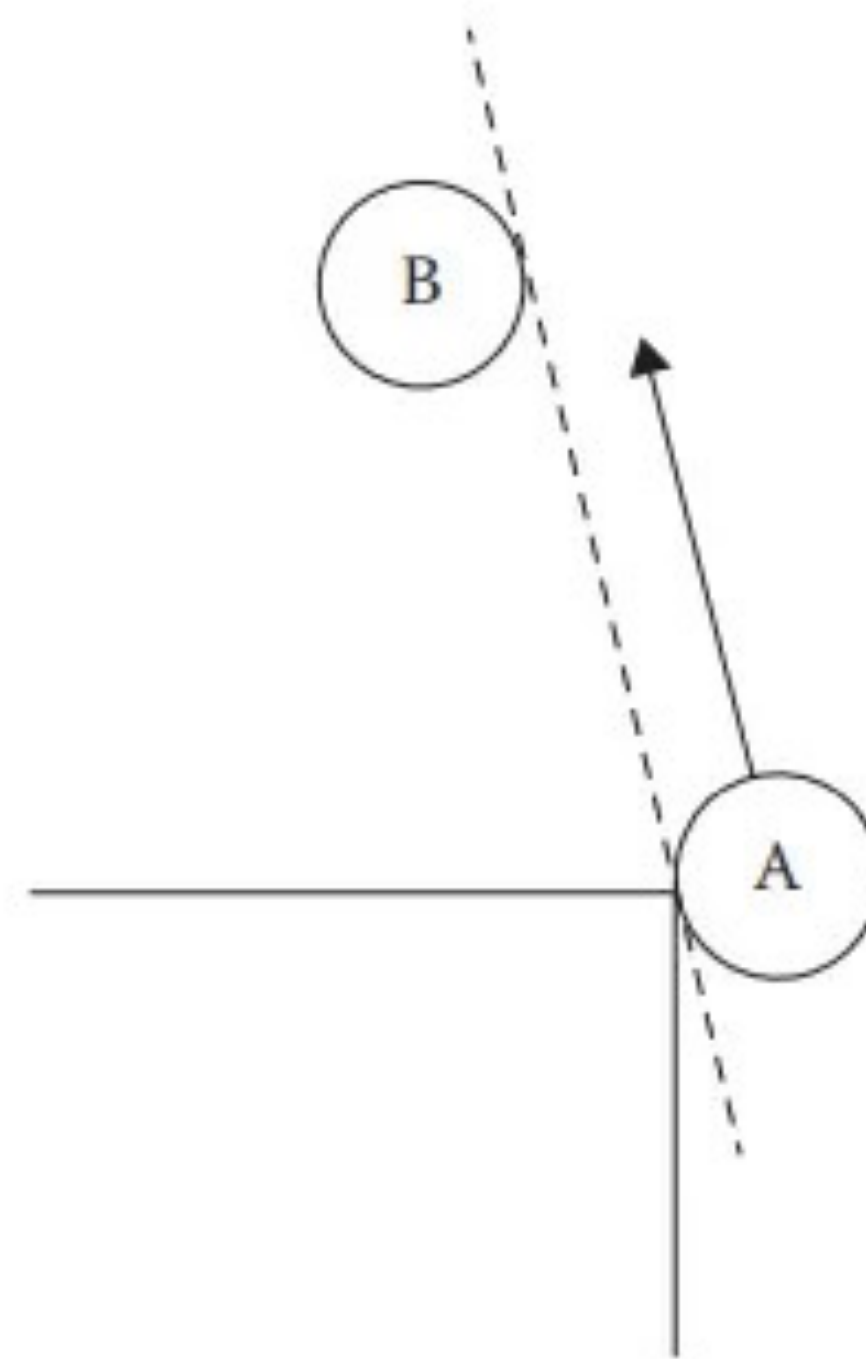
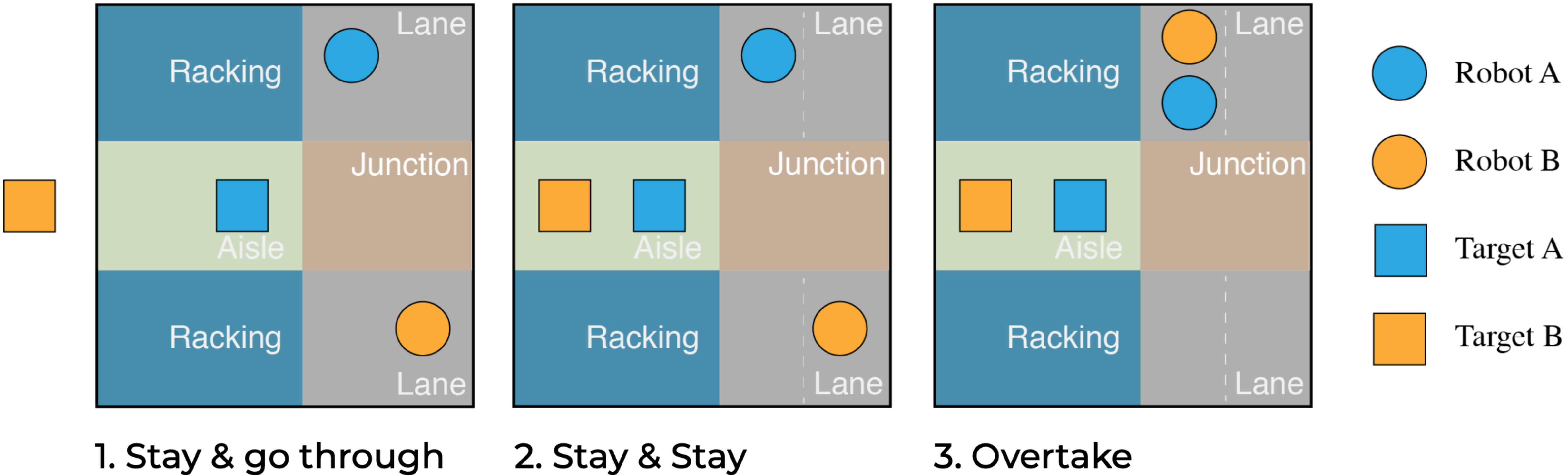


Figure 19.5

ORCA agent A is unable to continue turning around the corner without violating the sidedness constraint against B.

Comparison between ORCA and SAMP *2

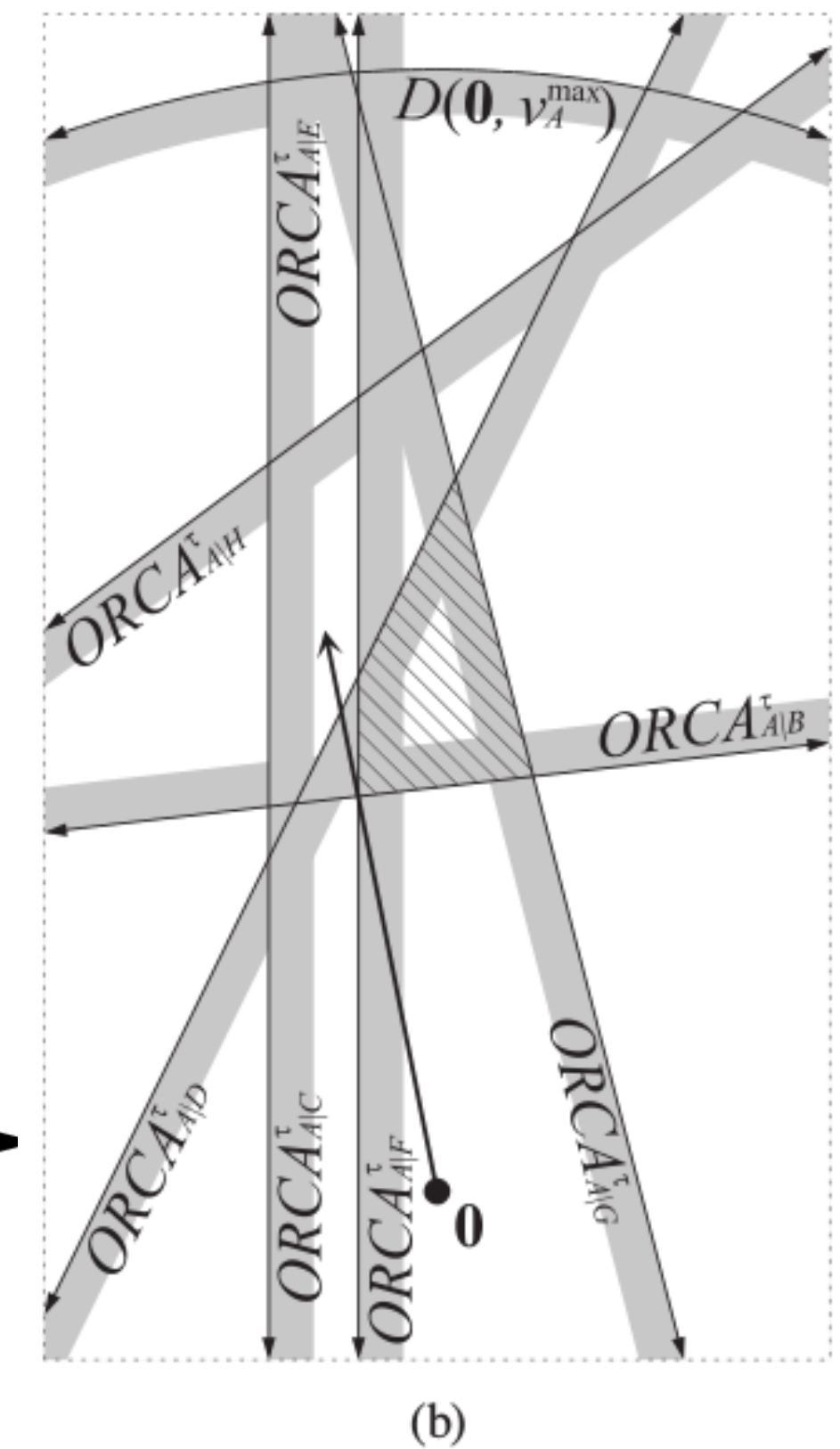
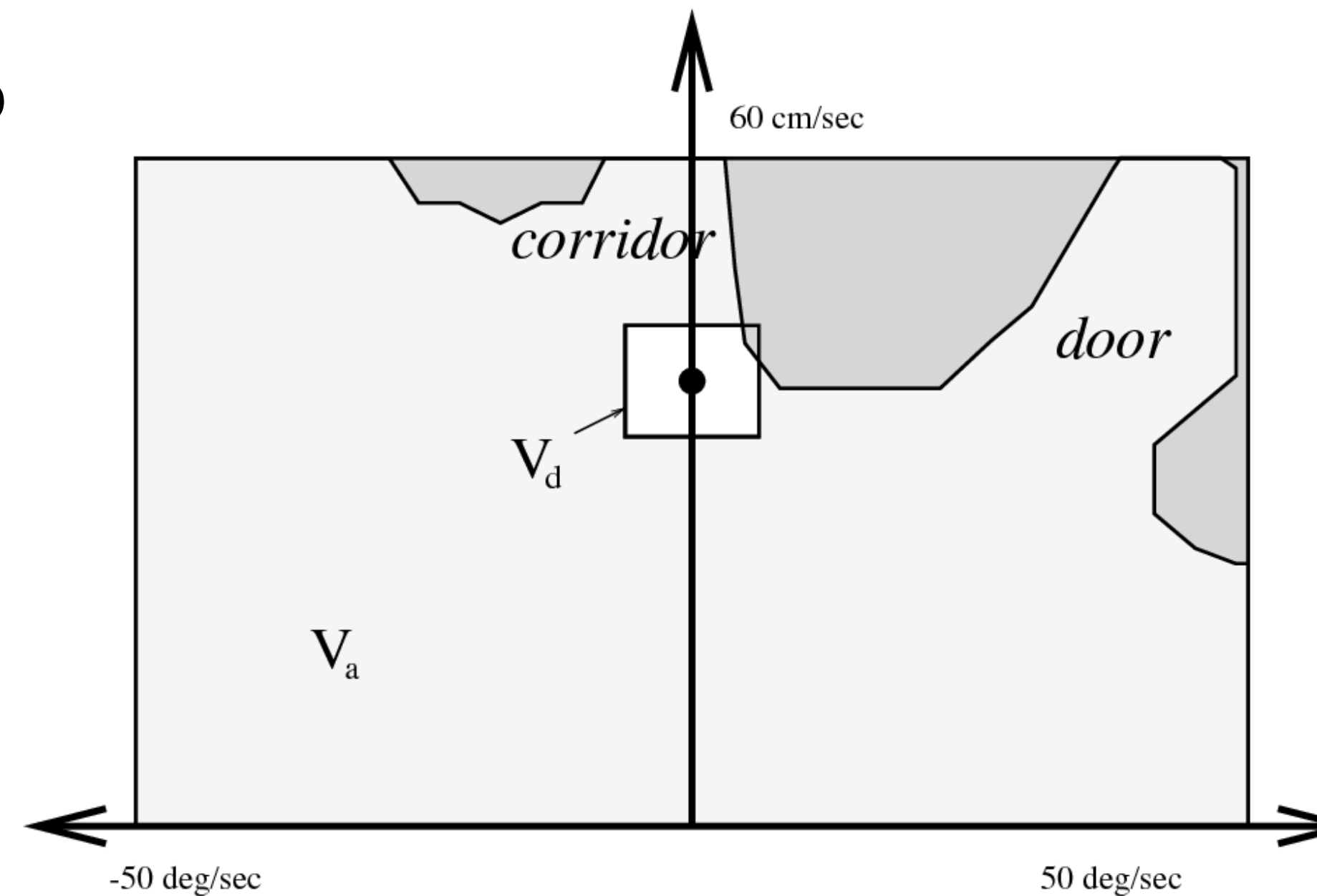
Interested scenarios, not yet implemented



Combine ORCA and SAMP

Action-based MPC

- Using ORCA to limit the search space to increase safety
- Turtlebot Input:
 - Linear velocity (m/s)
 - Angular velocity (rad/s)
- ORCA Output search space:
 - V_x (m/s)
 - V_y (m/s)



Potential Challenges

1. Mapping between two different coordinate
2. Computation load may increase
3. Discard too much search space may result in no feasible solution found
4. Perception accuracy