Paper Review 2

Lu, Hong 2020.8

西湖大學
WESTLAKE UNIVERSITY

Contents

- Avoiding target congestion on the navigation of robotic swarms
- Decentralized 3D collision avoidance for multiple UAVs in outdoor environments
- Decentralized navigation of multiple agents based on ORCA and model predictive control
- Dynamically Constrained Motion Planning Networks for Non-Holonomic Robots
- Collision avoidance for aerial vehicles in multi-agent scenarios

AVOIDING TARGET CONGESTION ON THE NAVIGATION OF ROBOTIC SWARMS

- Purpose
  - Reduce the congestion when agents in the swarm heading to the same target.
- Methodology
  - Probability Finite Machine
  - Divide scenario into two regions
  - Combine two above algorithms

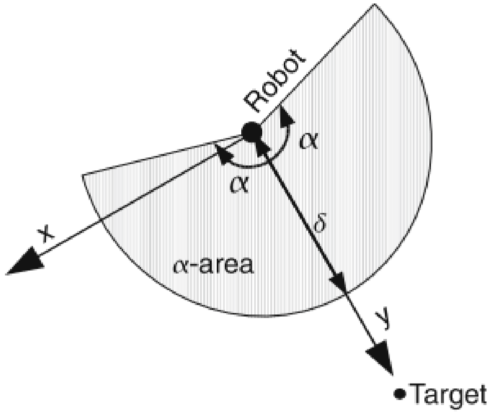


Fig. 3 Sensing area (α-area) considered by a robot to change its state

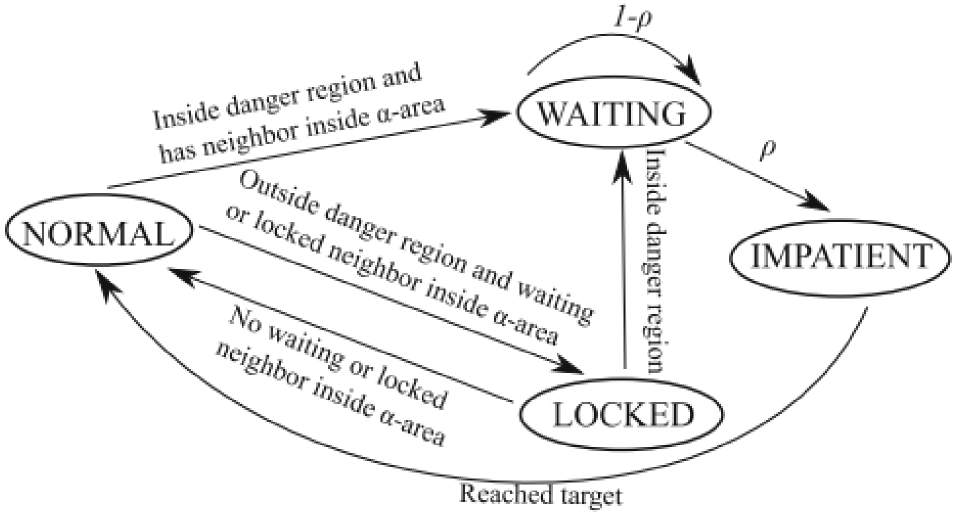


Fig. 1 Probabilistic finite state machine of the PCC algorithm

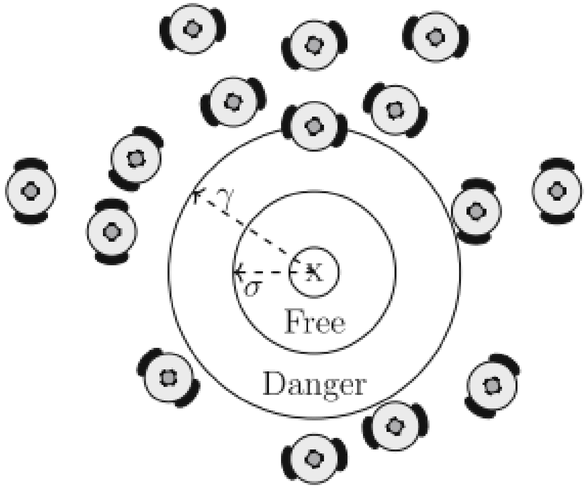


Fig. 2 *Free* and *danger* regions. "X" indicates the position of the target

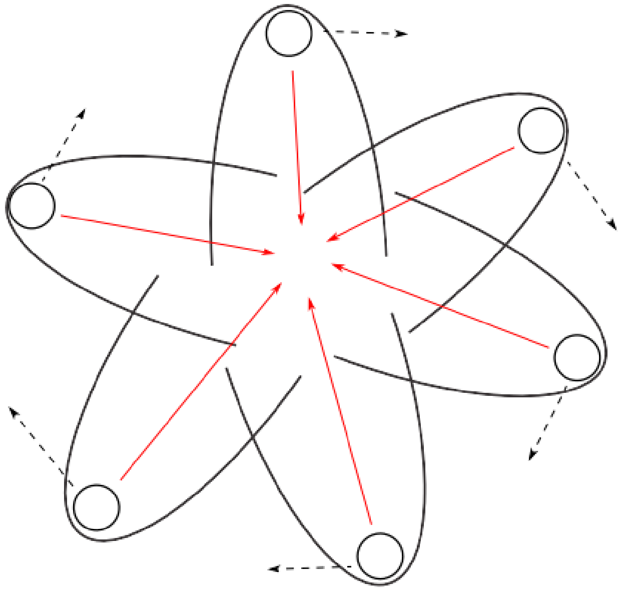
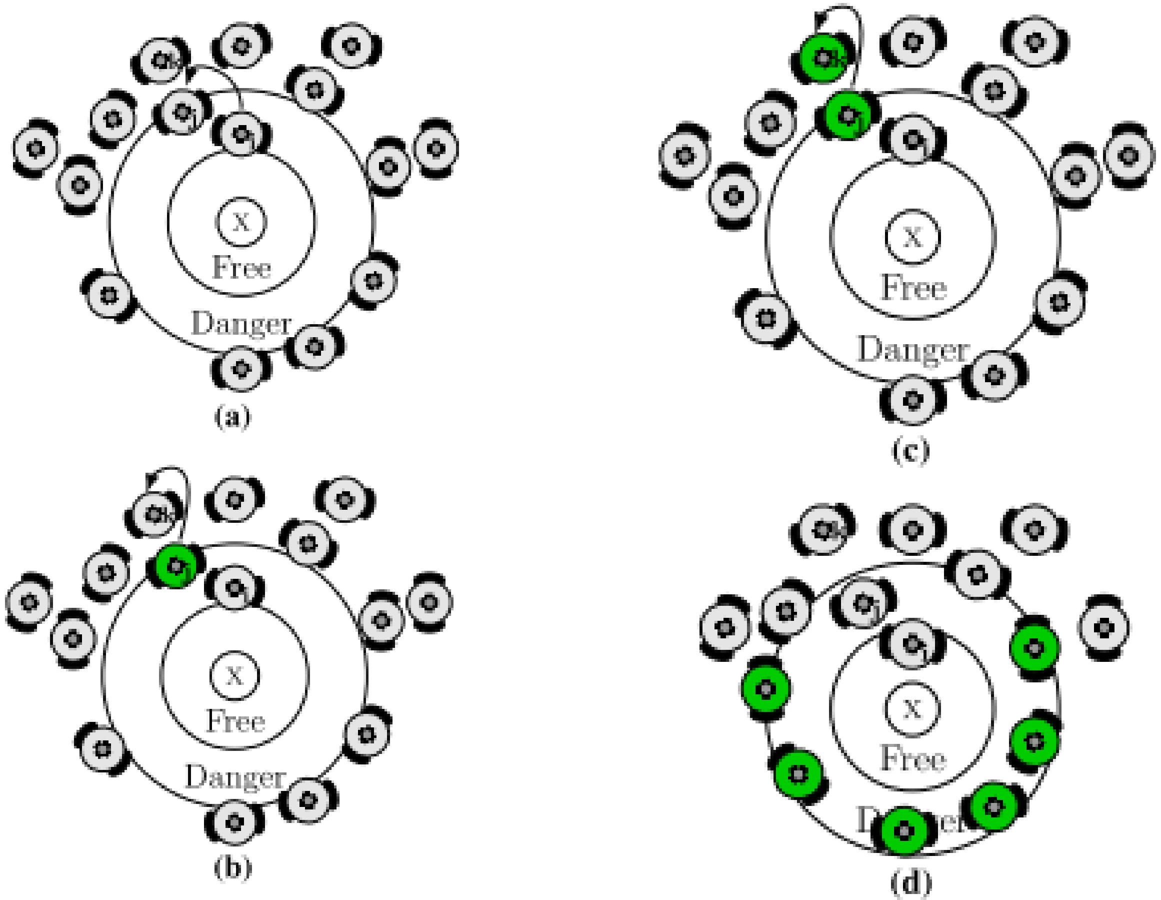
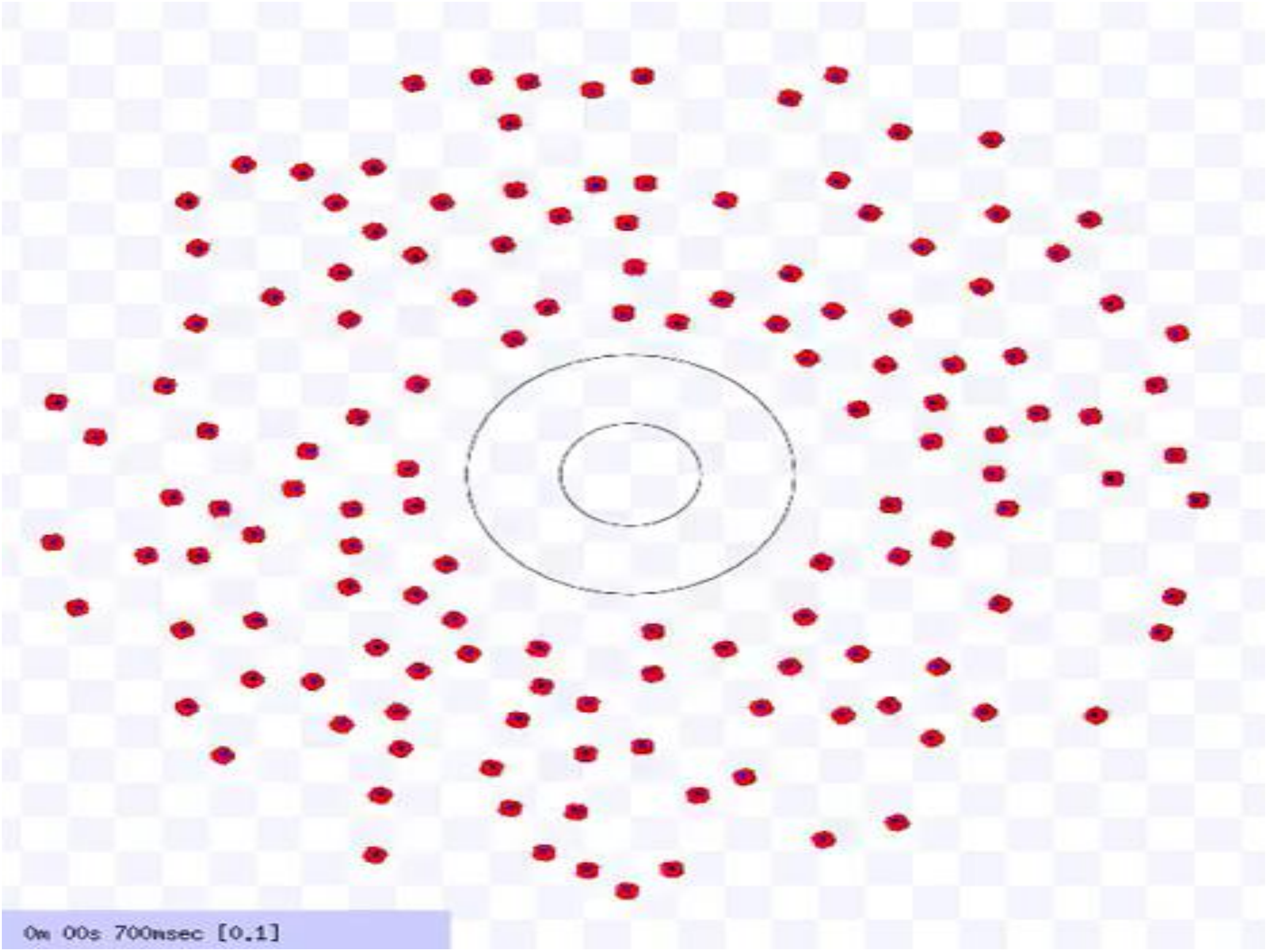


Fig. 10 ORCA reaches an equilibrium state in the common target problem

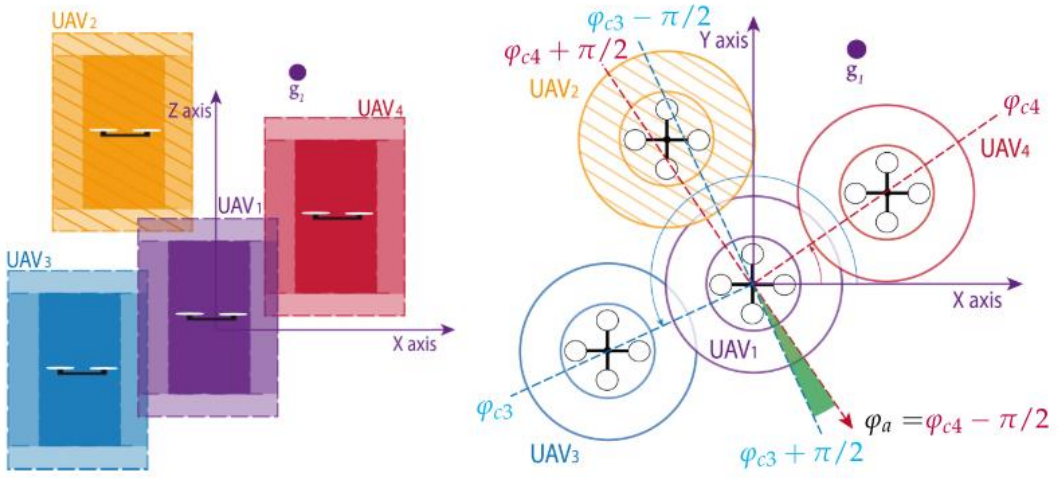



0m 00s 700msec [0,1]

DECENTRALIZED 3D COLLISION AVOIDANCE FOR MULTIPLE UAVS IN OUTDOOR ENVIRONMENTS

- Purpose
  - Extend their former 2D SWAP (Safety-enhanced avoidance policy) algorithm

- Methodology
  - Model the quadrotors as cylinders
    - Check collision efficiently
    - Make sense due to the downwash effect on other vehicle



(a) (b)

Algorithm 1 3D-SWAP for each UAV i

Input: Pointcloud from local sensors, current position $\mathbf{p}'_i$, goal position $\mathbf{g}_i$

while $\mathbf{g}_i$ not reached **do**
  Send position $\mathbf{p}'_i$ to neighbors
  $COD \leftarrow initCOD()$
  for all UAV j within communication range **do**
    Receive its position $\mathbf{p}'_j$
    $(\rho'_j, \varphi'_j, z'_j) \leftarrow transformToCylindrical(\mathbf{p}'_j)$
    $COD \leftarrow UpdateCOD(\rho'_j, \varphi'_j, z'_j)$
  end for
  for $m = 1$ **to** $m = M$ **do**
    Extract point m from local pointcloud
    $COD \leftarrow UpdateCOD(\rho_m, \varphi_m, z_m)$
  end for
  $(s_{xy}, s_z) \leftarrow computeState(COD)$
  $\mathbf{d}^g = \mathbf{g}_i - \mathbf{p}'_i$
  if s_{xy} is `xy-free` **then**
    $\angle \mathbf{v}_{xy}^{ref} = \angle \mathbf{d}_{xy}^g$ and $||\mathbf{v}_{xy}^{ref}|| = computeRefSpeed(||\mathbf{d}_{xy}^g||, v_{max})$
  else if s_{xy} is `rendezvous` **then**
    $\angle \mathbf{v}_{xy}^{ref} = \varphi_a$ and $||\mathbf{v}_{xy}^{ref}|| = v_a$
  else if s_{xy} is `xy-blocked` **then**
    $\mathbf{v}_{xy}^{ref} = \mathbf{0}$
  end if
  if s_z is `z-free` **then**
    $v_z^{ref} = \pm computeRefSpeed(d_z^g, v_{max})$, depending on whether $\mathbf{g}_i$ is above or below
  else if s_z is `z-blocked` **then**
    $v_z^{ref} = 0$
  end if
  $\mathbf{v}^{ref} \leftarrow (\mathbf{v}_{xy}^{ref}, v_z^{ref})$
  Send $\mathbf{v}^{ref}$ to velocity controller
end while

Simulation with 4 UAVs

- Goal of each UAV represented with a sphere of the same color.
- Reserved cylinders and collision cylinder are represented.
- Direction to the goal is represented by blue arrow.
- Avoidance direction is represented by red arrow (in case of conflict).

3D view

- Useful to visualize the behavior in altitude.

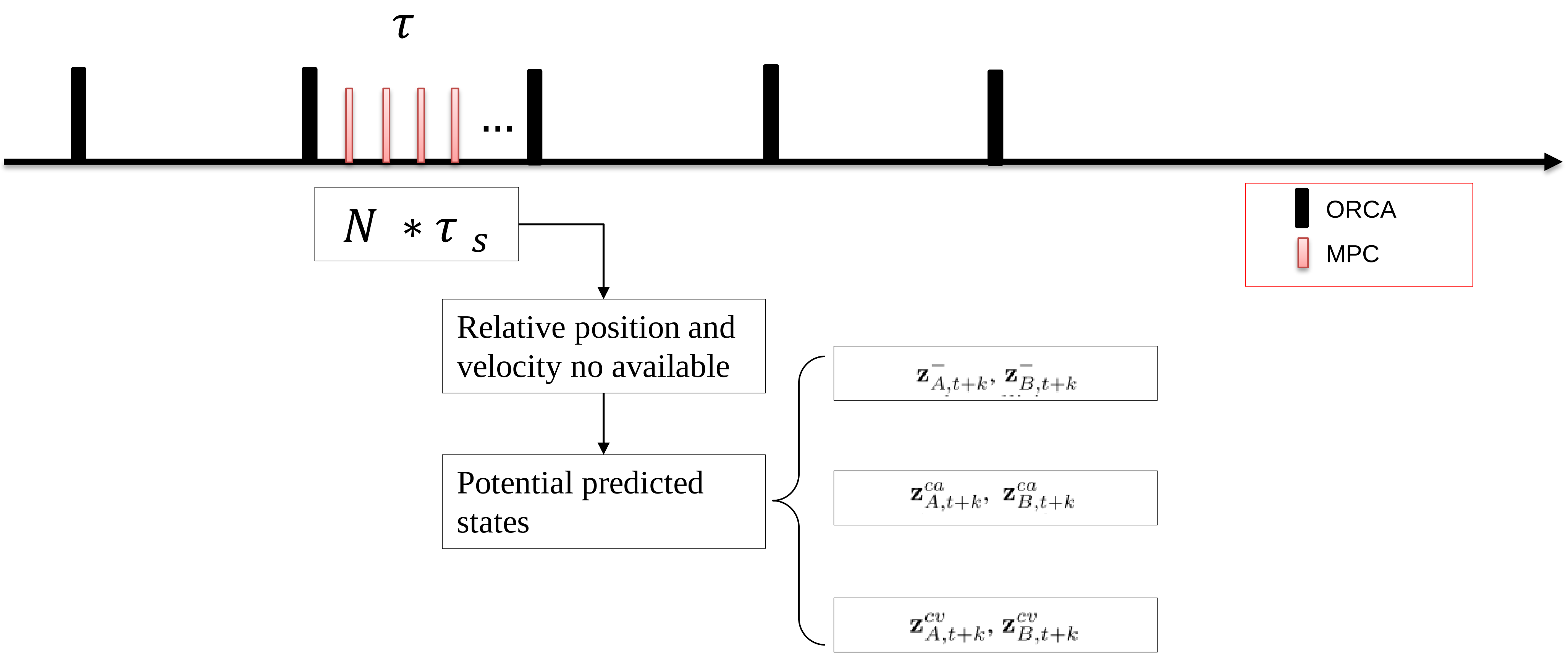
Top view

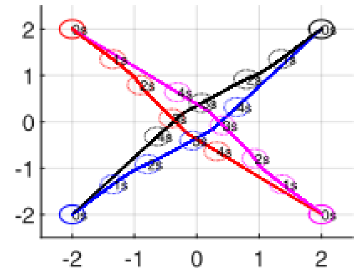
- Useful to visualize the avoidance maneuvers on the xy-plane.

DECENTRALIZED NAVIGATION OF MULTIPLE AGENTS BASED ON ORCA AND MODEL PREDICTIVE CONTROL

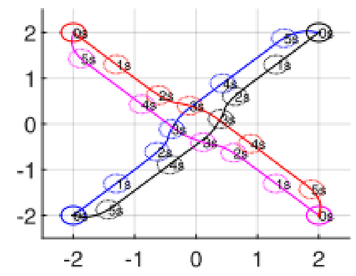
- Purpose
  - navigation
- Methodology
  - ORCA-MPC
    - ORCA calculates a permitted velocity set
    - MPC calculates optimal control input within the safety zone
      - Potential predicted state choices

To derive the optimal solutions, one should run the ORCA algorithm at each time $t + k$ for $1 \leq k \leq N$ by using the actual relative positions and velocities at $t + k$. Apparently,

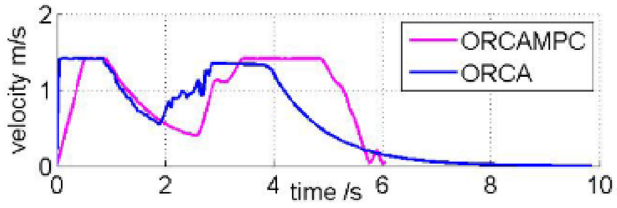




(a) Motion trajectories of 4 agents using ORCA.



(b) Motion trajectories of 4 agents using ORCA-MPC.



(c) Velocities of one of the four agents using two algorithms

Fig. 6. Comparison between ORCA and ORCA-MPC. In (a) and (b), colored circles represent different agents, and dotted lines are their trajectories.

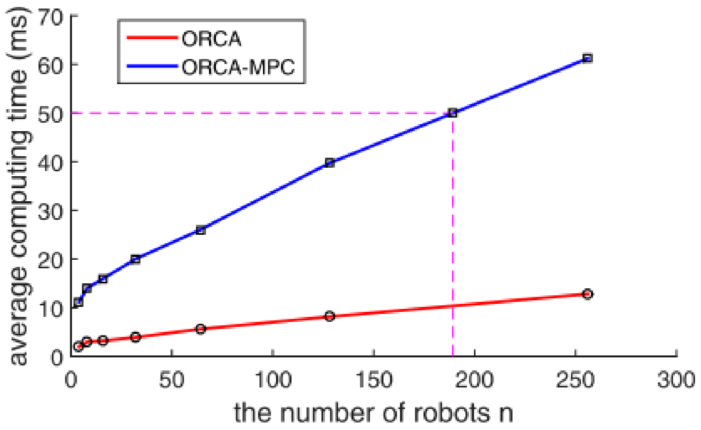


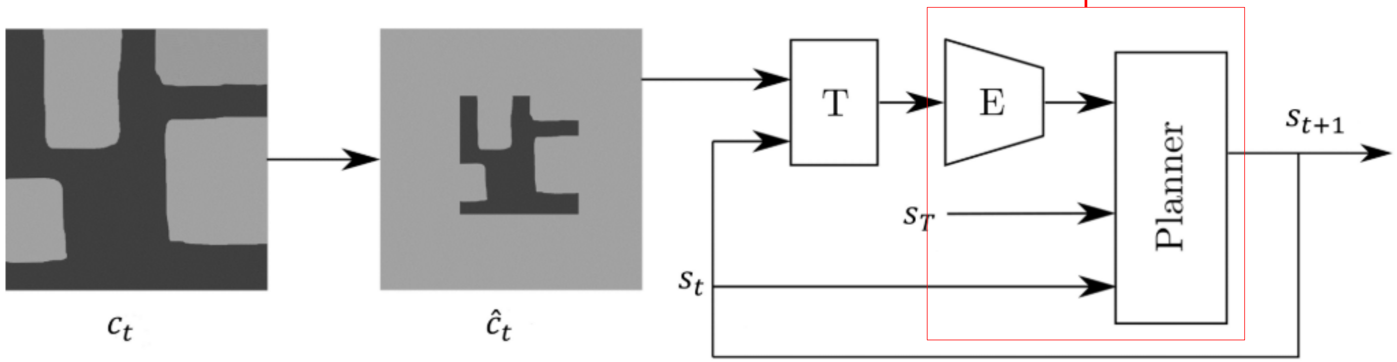
Fig. 7. Average computing time per agent per execution vs the number of agents (The predict time horizon is $N = 10$).

DYNAMICALLY CONSTRAINED MOTION PLANNING NETWORKS FOR NON-HOLONOMIC ROBOTS

- Purpose
  - Real-time planning for kinematically constrained vehicle
- Methodology
  - Dynamic Motion Planning Networks
    - Transformer (egocentric cost-map, size control as normalization)
    - Encoder
    - Planner
  - Training data
    - Supervised learning using expert data (RRT*)

$$L(\theta) = \frac{1}{N\ T} \sum_{j=1}^{N} \sum_{t=1}^{T-1} ||s_{j,t} - \hat{s}_{j,t}||^2$$

minimize



a training tuple $(s_t, s_T, \hat{c}_t, s_{t+1})$

Fig. 3: The graph describes the flow of inputs and outputs for the planner. x_t, x_{t+1} and x_g represents the current, next and target positions respectively. x_g is the sub-goal position from the global plan. C_t and, $\hat{C}_t$ is the costmap before and after padding respectively. The T block centers the padded costmap, $\hat{C}_t$, with respect to the robot position x_t. E block consists of convolution networks that encode the costmap into latent space vectors. The latent space representation of the costmap, the current robot and goal position are passed to the Planner node to generate the new target point.

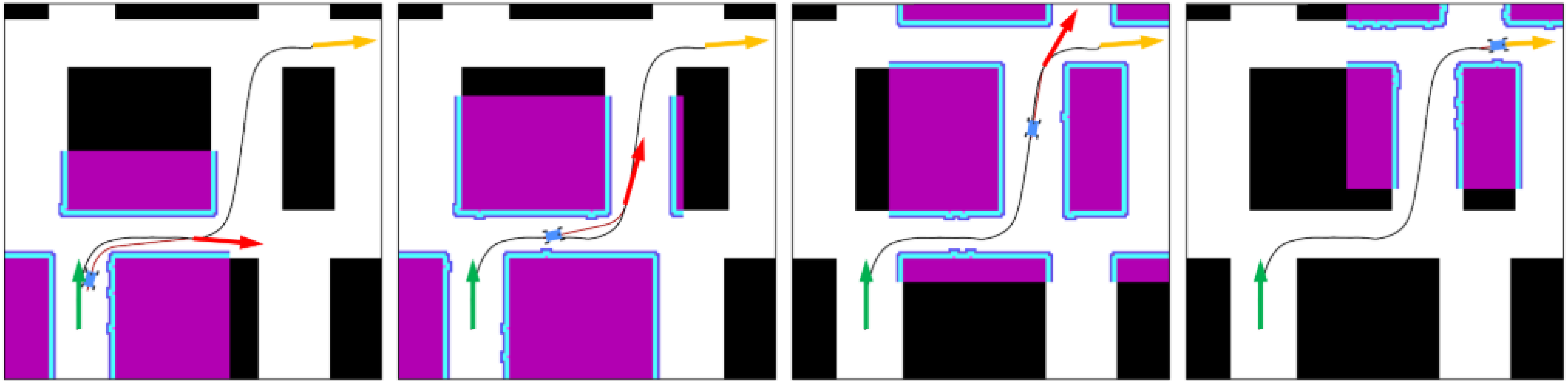
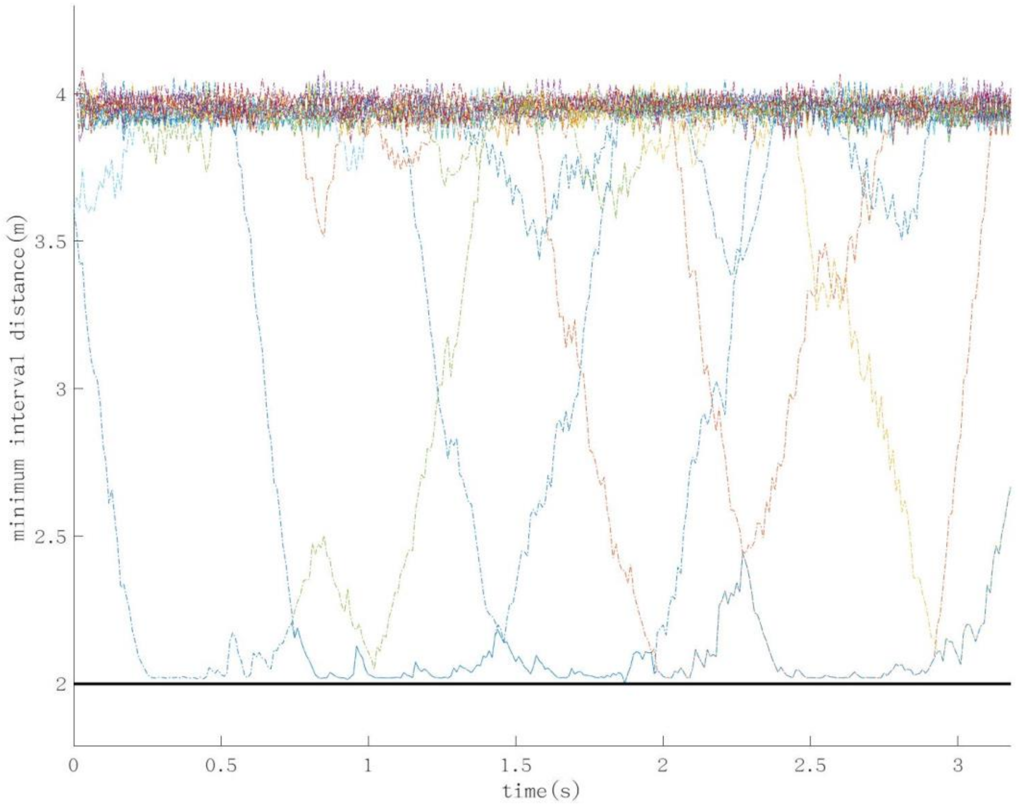
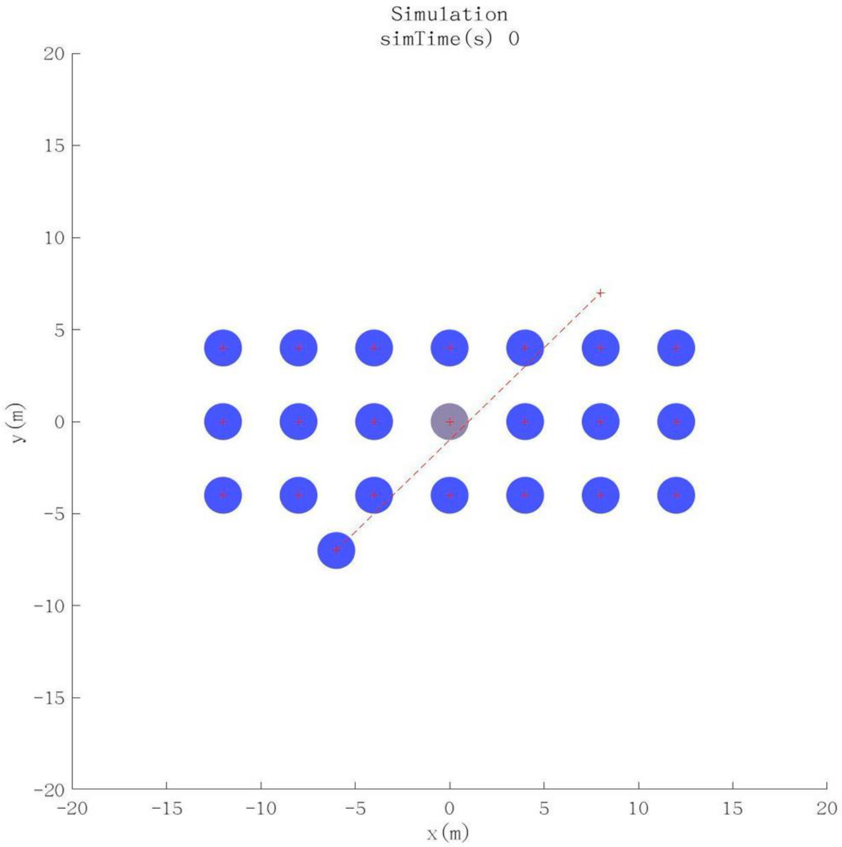


Fig. 4: For a given start (green arrow) and goal (orange arrow) position, the plan generated by the Dynamic MPNet (red path) for a given sub-goal (red arrow). The black trajectory is the global plan. The colored region represents the local costmap used by the Dynamic MPNet.


minimum interval distance (m)
time(s)


Simulation
simTime(s) 0
y(m)
x(m)

THANK YOU

西湖大學
WESTLAKE UNIVERSITY