Summary of: Motion planning for formations of mobile robots

Written by: TD Barfoot and CM Clark
Presented by: Kalliyan Velasco

Barfoot, T.D. & Clark, Christopher. (2004). Motion Planning for Formations of Mobile Robots. Robotics and Autonomous Systems. 46. 65-78. 10.1016/j.robot.2003.11.004.

Outline

- Motivation and background
- Assumptions and capabilities
- Methods
- Derivations
- Simulation results
- Future work

Motivation

Problem:

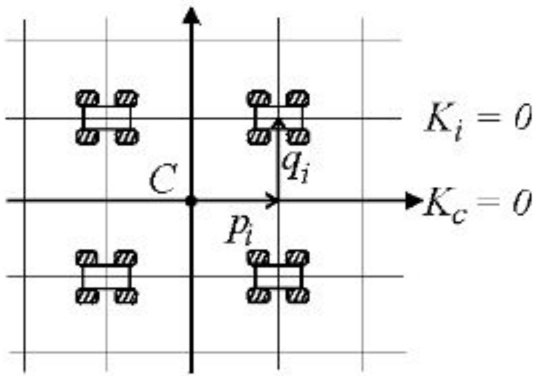
- How can formations of robots be maintained during motion?

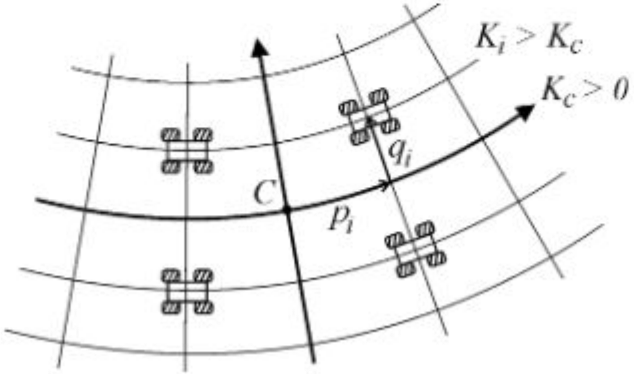
Applications:

- Space exploration - relocating distributed array of instruments (e.g., seismology, meteorology, three dimensional vision)
- Underwater robotics - terrain mapping

Model

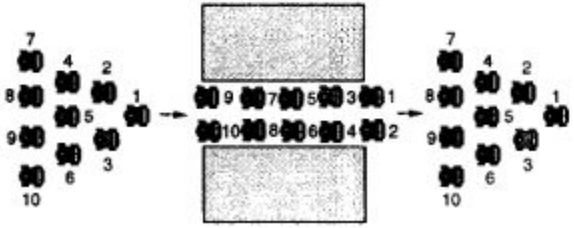
Main Idea: Use geometry to model network of robots as a single large robot. Constrain movement to a curvilinear coordinate system.





Relevant Background Literature

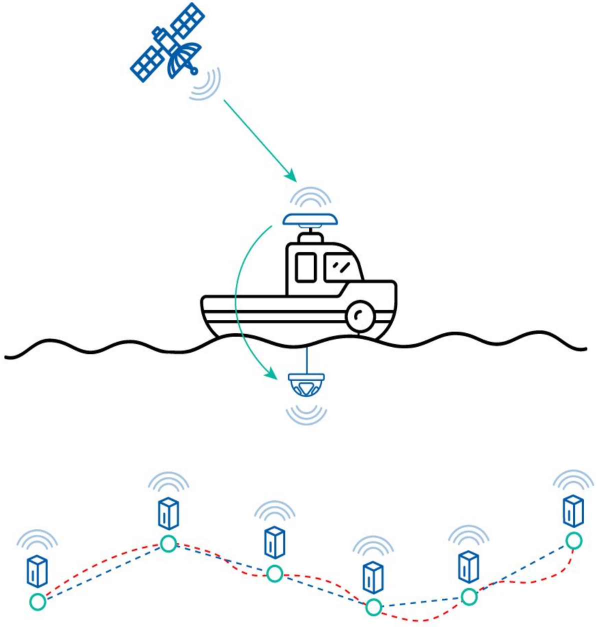
- Other methods for formations: behaviour-based, potential field, virtual structures, leader-follower, and neural networks
- Most similar to: [Control of changes in formation for a team of mobile robots (Desai 1999)]
    - Uses a reference trajectory and defines the motion of each individual robot relative to this trajectory
    - Does not use a curvilinear coordinate system



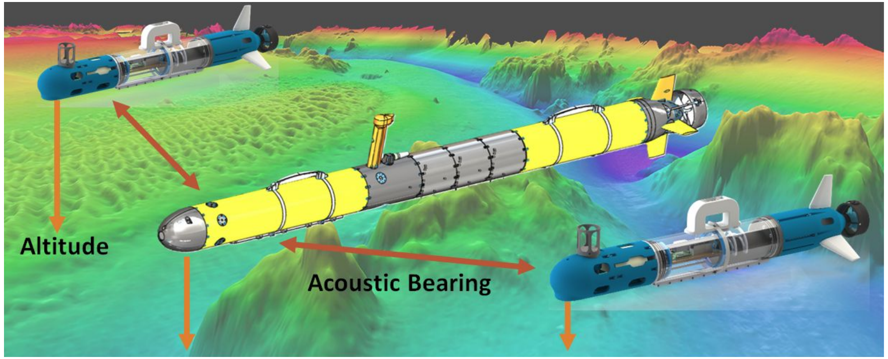
Assumptions

- Nonholonomic robots
  - Robot cannot move in certain directions due to mechanical constraints
  - From Wiki: **holonomic constraint**…does not depend on the velocities or any higher-order derivative with respect to *t*
- Each robot knows its own global location
  - Cannot sense the locations of other robots
- Each robot knows the planned motion of the reference trajectory
  - Can communicate with leader, but not each other
- Known initial conditions (does not allow from random initial conditions)
- Obstacle avoidance not addressed - handled by reference trajectory planner

Application: Underwater Robotics



Localization using USBL



Multi-agent terrain mapping

Capabilities

- Applicable to nonholonomic robots
- Reference trajectory can be computed in real time and the shape of the formation can be described independently of the reference trajectory
- Flexible approach - robot formations can change over time
  - Change formation scale to collect data at different resolutions
  - Formation wide for data collection, becomes narrow to fit between two obstacles

Robot Model

Robot model in Cartesian space:

$$\dot{x}_c = v_c \cos\theta_c, \quad \dot{y}_c = v_c \sin\theta_c, \quad \dot{\theta}_c = v_c K_c$$

Nonholonomic constraint:

$$-\dot{x}_c \sin\theta_c + \dot{y}_c \cos\theta_c = 0$$

Curvature and speed constraints:

$$|K_c| \leq K_{c,max}, \qquad |v_c| \leq v_{c,max}$$

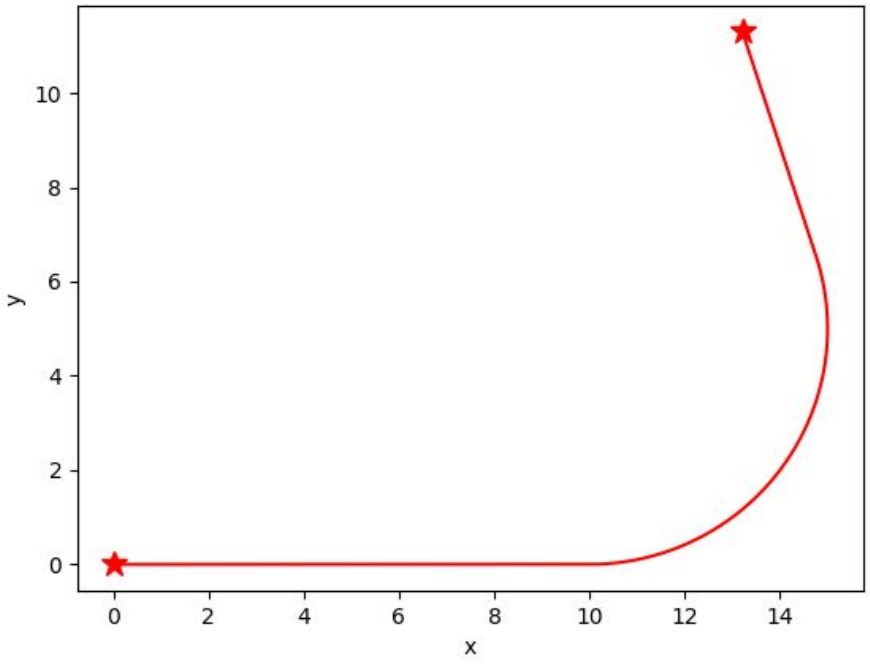
Method

1. Get a reference trajectory
2. The reference trajectory is communicated with each of the individual robots

Assume known:

$$\begin{bmatrix} x_c(t) & y_c(t) & \theta_c(t) \end{bmatrix}^T$$

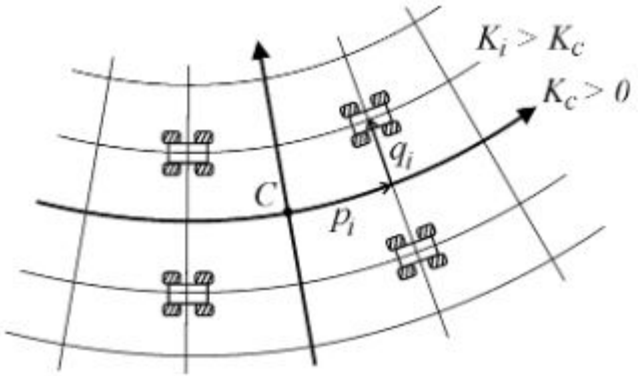
$$\begin{bmatrix} v_c(t) & K_c(t) \end{bmatrix}^T$$

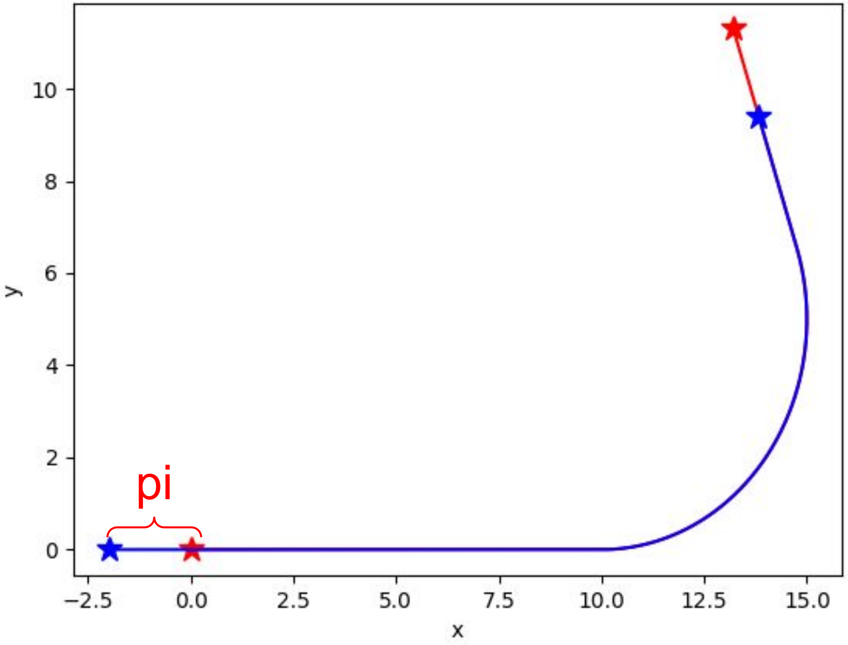


Method

3. Each robot gets reference trajectory, taking pi shift into account

$$s_i(t) = d_c(t) + p_i$$





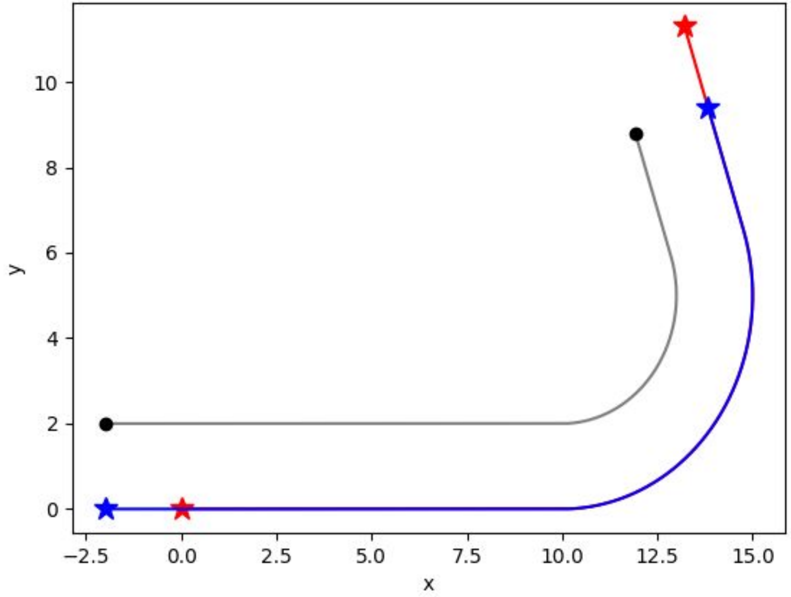
Method

4. Each robot calculates own trajectory, taking qi into account

$$v_i(s_i) = v_c(s_i)\Big(1 - q_i K_c(s_i)\Big)$$

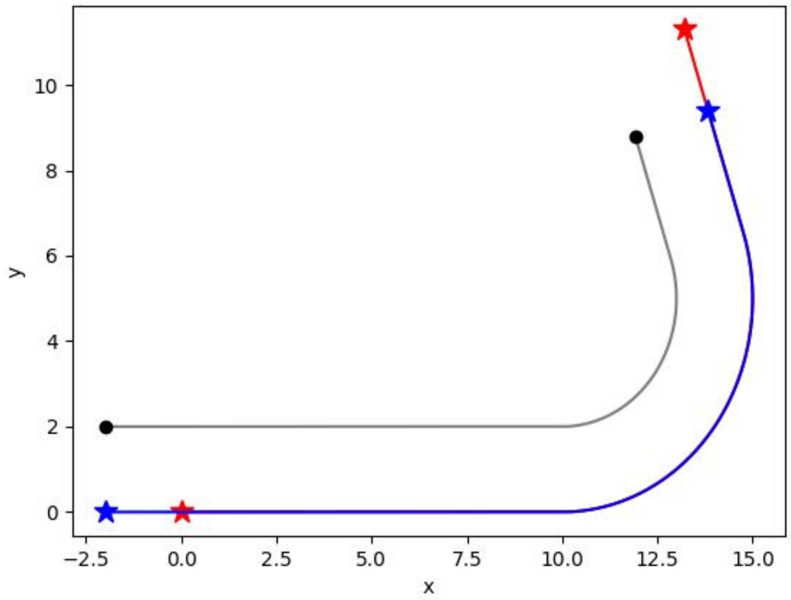
$$K_i(s_i) = \frac{K_c(s_i)}{1 - q_i K_c(s_i)}$$

$$\dot{x}_i = v_i \cos\theta_i, \quad \dot{y}_i = v_i \sin\theta_i, \quad \dot{\theta}_i = v_i K_i$$



Method

5. Each robot uses a feedback control loop to ensure it follows its planned trajectory

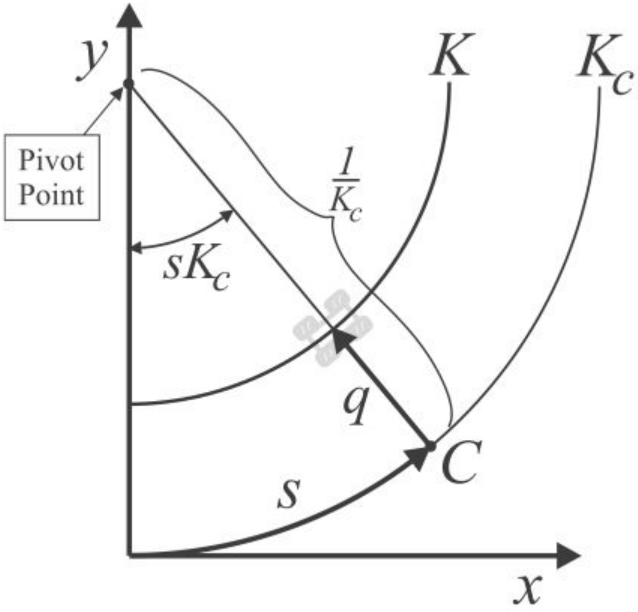


Derivations

Position of follower -

Curvilinear to Cartesian

$$
\begin{aligned}
x &= \left(\frac{1}{K_c} - q\right)\sin(sK_c) \\
y &= \frac{1}{K_c} - \left(\frac{1}{K_c} - q\right)\cos(sK_c)
\end{aligned}
$$



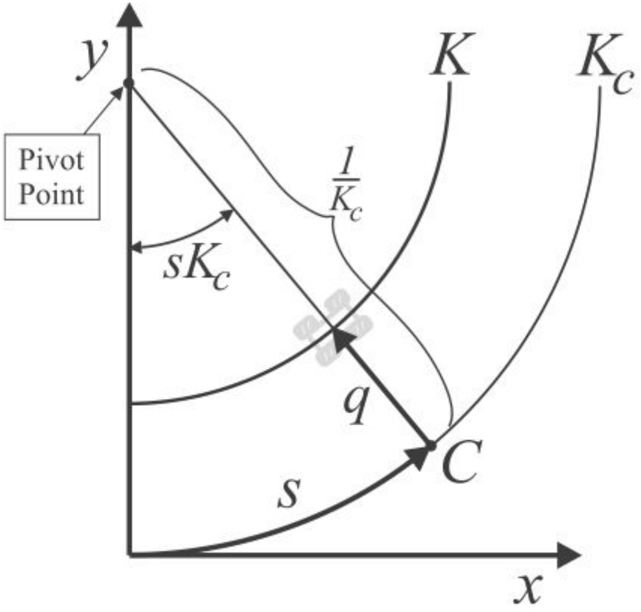
Derivations

Take derivatives - K is constant

$$
\begin{aligned}
x &= \left(\frac{1}{K_c} - q\right)\sin(sK_c) \\
y &= \frac{1}{K_c} - \left(\frac{1}{K_c} - q\right)\cos(sK_c)
\end{aligned}
$$

$$
\begin{aligned}
\dot{x} &= -\dot{q}\sin(sK_c) + \dot{s}\,(1 - qK_c)\cos(sK_c) \\
\dot{y} &= \dot{q}\cos(sK_c) + \dot{s}\,(1 - qK_c)\sin(sK_c)
\end{aligned}
$$

$$
\begin{aligned}
\ddot{x} &= -\left(\ddot{q} + \dot{s}^2 K_c(1 - qK_c)\right)\sin(sK_c) + \left(\ddot{s}(1 - qK_c) - 2\dot{s}\dot{q}K_c\right)\cos(sK_c) \\
\ddot{y} &= \left(\ddot{q} + \dot{s}^2 K_c(1 - qK_c)\right)\cos(sK_c) + \left(\ddot{s}(1 - qK_c) - 2\dot{s}\dot{q}K_c\right)\sin(sK_c)
\end{aligned}
$$



Derivations - Velocity (Dynamic Formations)

Identity to use:

$$\dot{x}^2 + \dot{y}^2 = \dot{q}^2 + \dot{p}^2(1 - qK_\mathrm{c})^2$$

Plug in derivatives:

$$v = (\dot{x}^2 + \dot{y}^2)^{1/2} = (\dot{q}^2 + \dot{p}^2(1 - qK_\mathrm{c})^2)^{1/2} = Qv_\mathrm{c}$$

$$Q = \left(\left(\frac{dq}{ds} \right)^2 + (1 - qK_c)^2 \right)^{\frac{1}{2}}$$

Derivations - Velocity (Static Formations)

Velocity, dynamic:

$$v = (\dot{x}^2 + \dot{y}^2)^{1/2} = (\dot{q}^2 + \dot{p}^2(1 - qK_c)^2)^{1/2} = Qv_c$$

$$Q = \left(\left(\frac{dq}{ds}\right)^2 + (1 - qK_c)^2\right)^{\frac{1}{2}}$$

If static, no change in q:

$$\frac{d^2q}{ds^2} = \frac{dq}{ds} = 0$$

Velocity, static:

$$\boxed{v = (1 - qK_c)v_c}$$

Derivations - Curvature (Dynamic Formations)

Equation for curvature:

$$K = \frac{\dot{x}\ddot{y} - \ddot{x}\dot{y}}{(\dot{x}^2 + \dot{y}^2)^{3/2}},$$

Plug in derivatives:

$$K = \frac{K_c \dot{s}\dot{q}^2 + (1 - qK_c)(\dot{s}\ddot{q} - \ddot{s}\dot{q}) + K_c \dot{s}\left(\dot{q}^2 + \dot{s}^2(1 - qK_c)^2\right)}{\left(\dot{q}^2 + \dot{s}^2(1 - qK_c)^2\right)^{\frac{3}{2}}}$$

Derivations - Curvature (Dynamic Formations)

Divide numerator and denominator of curvature by sdot^3 to use:

$$\frac{d^2q}{ds^2} = \frac{\dot{s}\ddot{q} - \ddot{s}\dot{q}}{\dot{s}^3}$$

Curvature simplifies down to:

$$\boxed{K = \frac{1}{Q}\left(K_c + \frac{\left(1 - qK_c\right)\frac{d^2q}{ds^2} + K_c\left(\frac{dq}{ds}\right)^2}{Q^2}\right)}$$

Derivations - Curvature (Static Formations)

Curvature, dynamic:

$$K = \frac{1}{Q}\left(K_c + \frac{\left(1 - qK_c\right)\frac{d^2q}{ds^2} + K_c\left(\frac{dq}{ds}\right)^2}{Q^2}\right) \qquad Q = \left(\left(\frac{dq}{ds}\right)^2 + (1 - qK_c)^2\right)^{\frac{1}{2}}$$

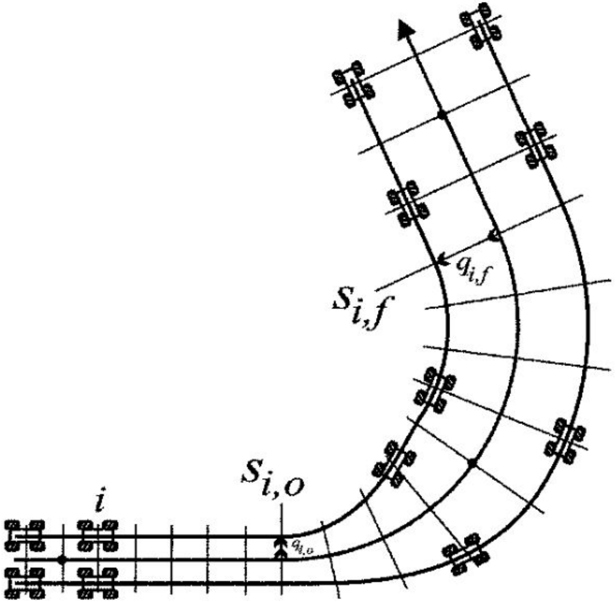
Again, if static, no change in q:

$$\frac{d^2q}{ds^2} = \frac{dq}{ds} = 0$$

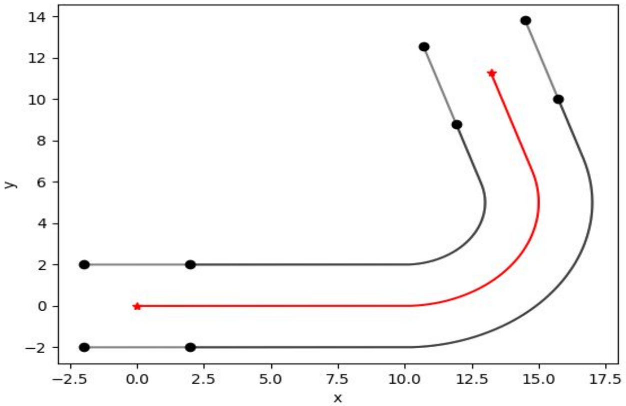
Curvature, static:

$$\boxed{K = \frac{K_c}{1 - qK_c}}$$

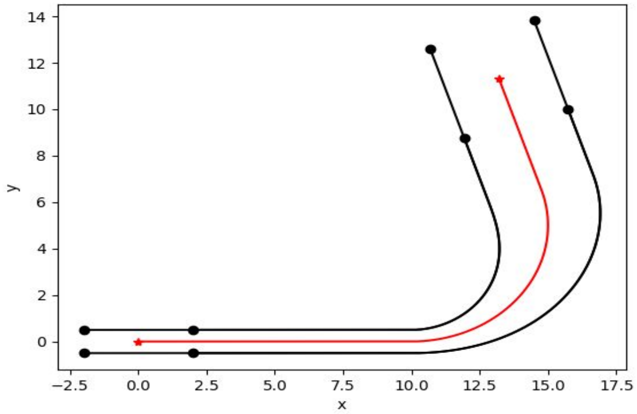
Simulation



Paper Formation

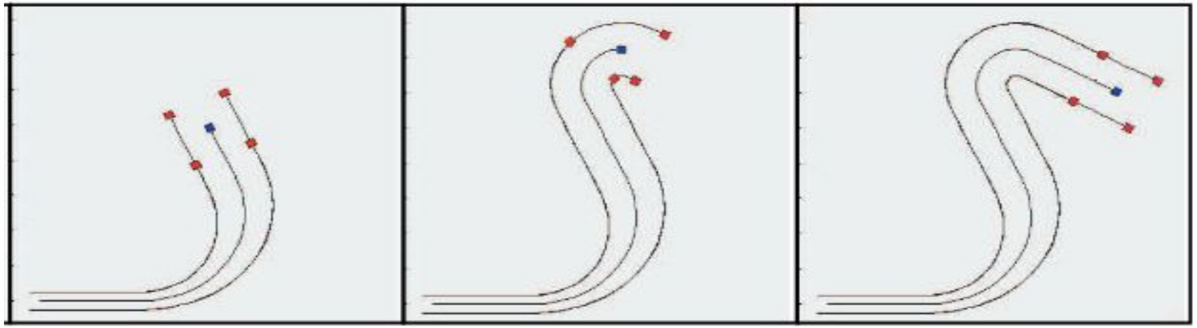


My Static Formation

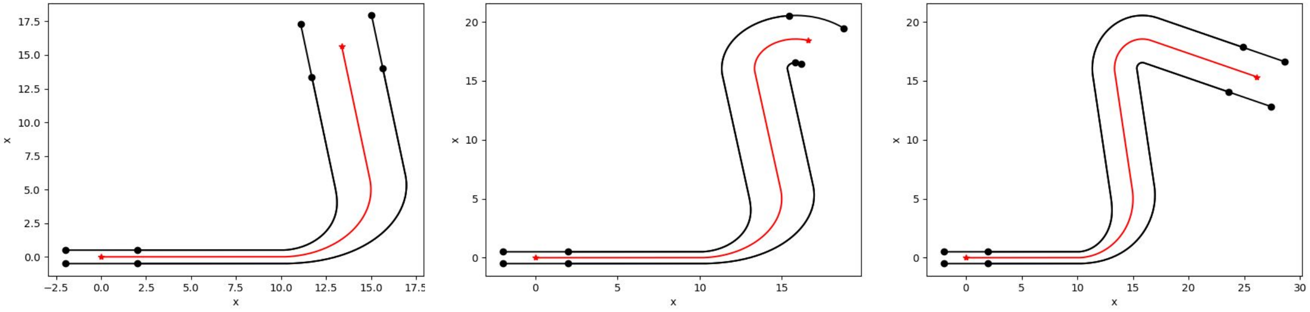


My Dynamic Formation

Simulation



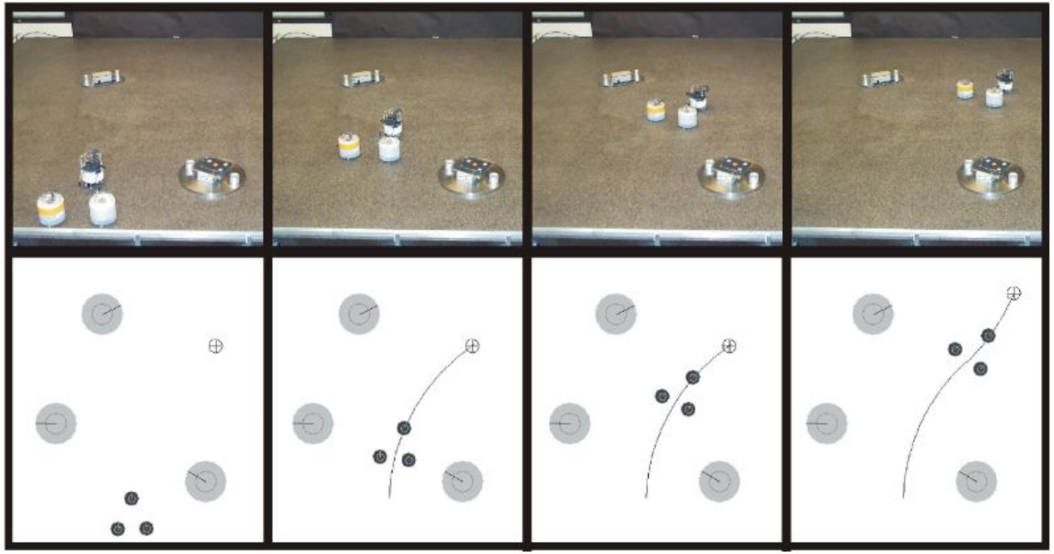
Paper



Mine

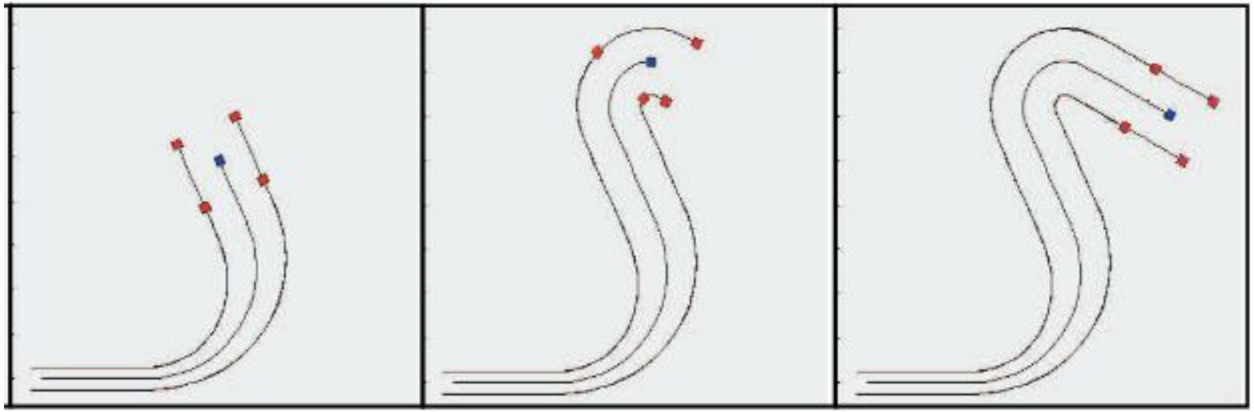
Hardware Results (Paper)

- Small disc shaped robots on 3mx2m table
- Overhead vision for position and velocity



When is this method not useful?

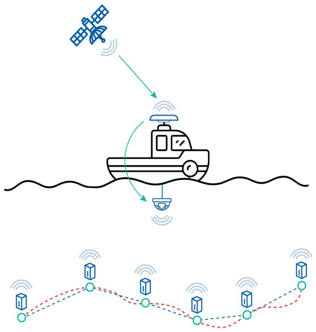
- Cooperative carrying situations where each robot holds the corner of a rigid object



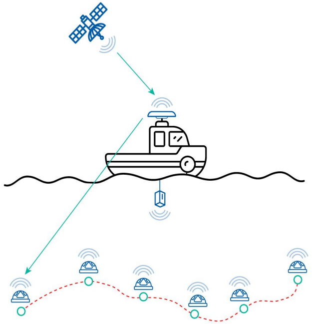
Future Work

[Decentralized Formation Control for Multiple Quadrotors under Unidirectional Communication Constraints (Roque 2020)]

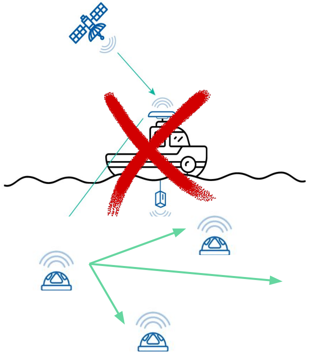
- Formation control using relative pose info, not global
- Apply to underwater (communication constraints)



Classic USBL



Inverted USBL



Inverted USBL - Decentralized

Other methods

A Survey of An Intelligent Multi-Agent Formation Control (Chen 2023)

Control Strategies	Advantages	Disadvantages
Leader–Follower	1. Simple design and implementation, easy to expand the formation 2. Better formation tracking performance 3. Simple communication topology	1. Formation stability is too dependent on the leader 2. No information interaction between the leader and the follower, poor robustness 3. Poor formation stability when there is a communication delay
Virtual Structure	High stability of intelligent agent formation	1. Poor flexibility, poor obstacle avoidance ability 2. Poor robustness against external interference 3. Excessive communication load can cause node failure
Behavior Based	Strong ability to respond to multiple targets and mission requirements in formations	1. Difficult to express the formation behavior by a mathematical model 2. Poor stability 3. Poor robustness in the face of an unknown environment because the formation behavior is loading first
Consensus Based	Enables formation control in limited and varying communication topologies	1. Simple intelligent agent model, only the first-order or second-order linear model is considered 2. Insufficient consideration for the motion form of a single agent, poor robustness
Intelligent Control	1. Less difficult to model 2. Robustness in dealing with location environment	1. Long computation time and high complexity 2. Calculation is a black box, difficulty to describe the control process mathematically and estimate accurately

Summary

Can maintain robot formations using simple geometry

Use this method if:

- You are working with nonholonomic robots
- All agents know their global position
- All agents know the reference trajectory

Don’t use this method if:

- You want a decentralized solution for formations
- You are carrying a rigid object

Thank you!