Intelligent Unmanned System Control Practice

Chapter 10 Swarm Collaboration and Game-Theoretic Confrontation

Author: Xunhua Dai, Central South University

Website: https://faculty.csu.edu.cn/daixunhua

Email: dai.xh@csu.edu.cn

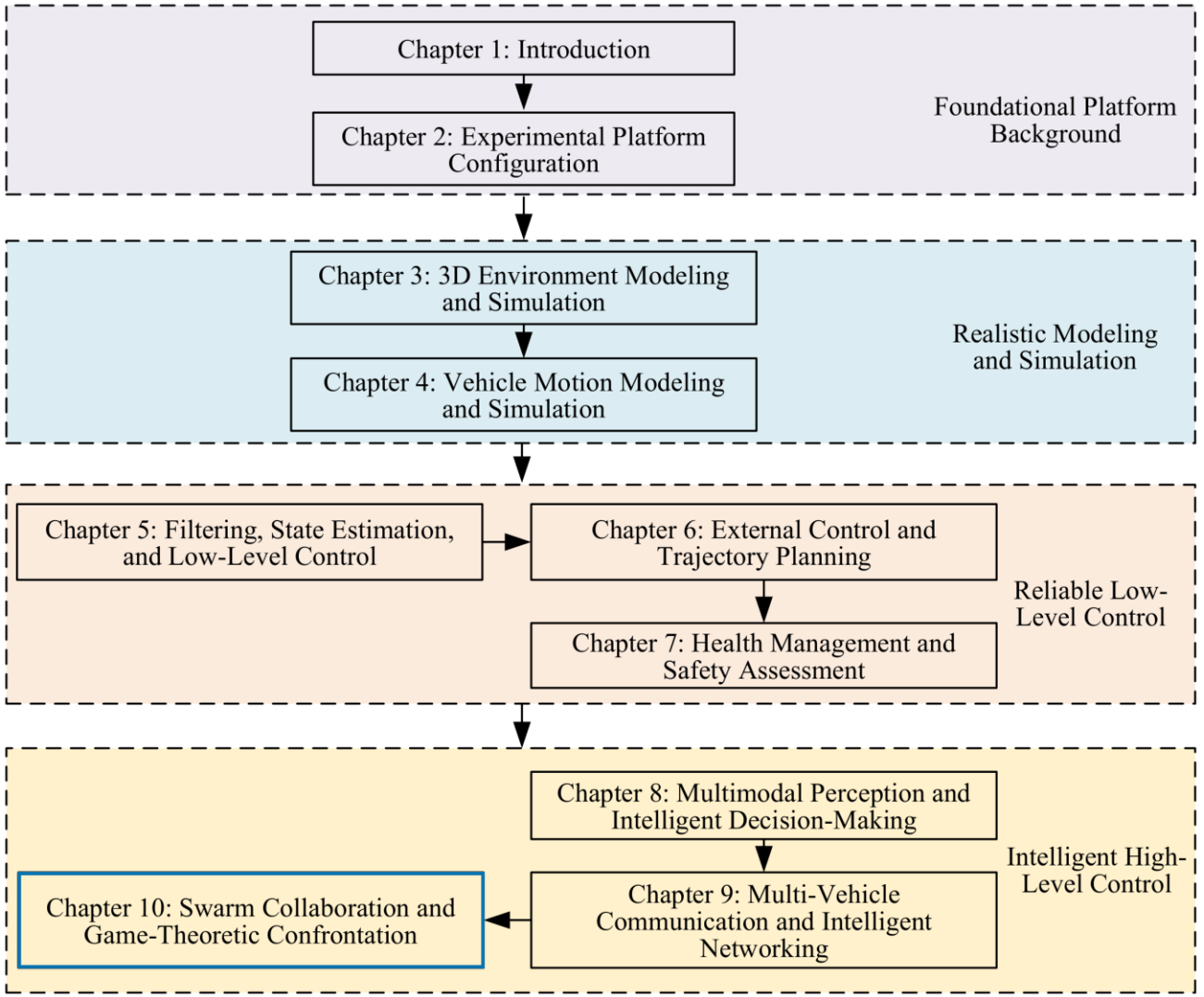
RflySim: https://rflysim.com

Companion Paper



Dai, Xunhua, Jinhu Tu, and Quan Quan. "RflySim ToolChain: a rapid development and validation toolchain for intelligent unmanned swarm systems." Journal of Systems Engineering and Electronics 36.4 (2025): 1077-1093.

Textbook is coming soon...



Chapter Learning Objectives

What You Can Do After Chapter 10

1. Explain information, decision, and execution flows to select a swarm control architecture
2. Compute consensus and formation commands, and explain convergence and safety conditions
3. Model sequential decisions and compare single-agent and multi-agent reinforcement learning (RL)
4. Connect RflySim interfaces and assess error, separation, and data timeliness in experiments

Chapter Outline

Swarm Collaboration Fundamentals and Control Architectures

Chapter 10 Swarm Collaboration and Game-Theoretic Confrontation

10.1 Swarm Fundamentals and Architectures

10.2 Consensus and Formation Control

10.3 Game Theory and RL Fundamentals

10.4 Single-Agent RL Algorithms

10.5 Multi-Agent RL and Scaling

10.6 Technologies and Challenges

10.7 RflySim Framework and Interfaces

10.8 Experiments and Validation

10.9 Exercises and Classroom Discussion

3 Units in This Section

10.1.1 Background and Tasks of Swarm Collaboration

Textbook: Sec. 10.1.1

10.1.2 Swarm Control Architecture

Textbook: Sec. 10.1.2.1

10.1.3 Decomposition of Swarm Control Tasks

Textbook: Sec. 10.1.2.2

10.1.1 Background and Tasks of Swarm Collaboration: Inspection Task

Coordination in a Three-Vehicle Inspection Task

Corridor inspection: 4 m neighbor spacing, with no collisions throughout the task.



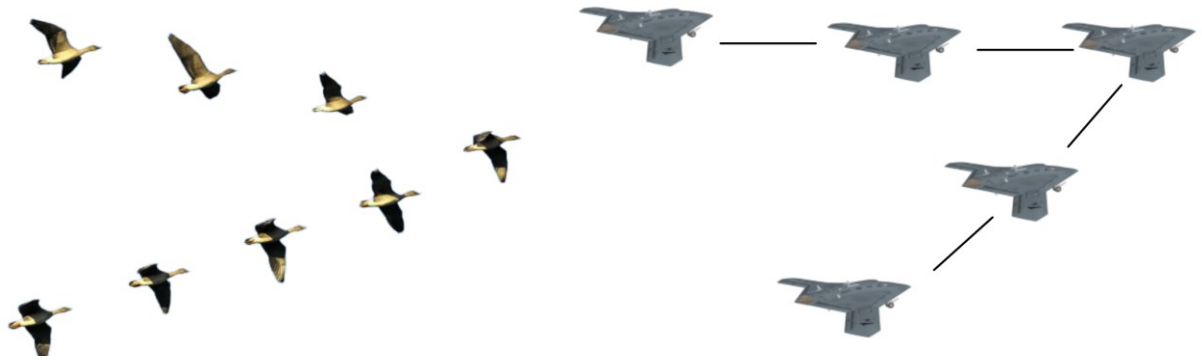
Individual control: each vehicle uses its own measurements to track a velocity or position command.

Swarm collaboration: use the shared task and other members' information to coordinate roles, motion, and fault response.

Collaboration adds defined roles and constraints to flying together.

10.1.1 Background and Tasks of Swarm Collaboration: Core Concepts

Biological Swarms and Multi-Agent Unmanned Systems



(a) Bird formation

(b) Unmanned aircraft formation

Fig. 10.1 From Biological Swarms to Unmanned Swarms

Swarm: a group system whose interacting members produce collective behavior.

Multi-agent system: multiple agents that sense, decide, act, and cooperate or compete.

Stigmergy: indirect coordination through information left in the environment, without direct messaging.

10.1.1 Background and Tasks of Swarm Collaboration: System Properties

Local Interaction and System Organization

System Property	Inspection Action	Result to Check
Self-organization	Neighbors share states to form specified relative relationships	Does formation error decrease?
Adaptation	Adjust routes and speeds after wind disturbances or obstacles	Does the task resume after a disturbance?
Fault Tolerance and Reconfiguration	Reassign areas after one vehicle leaves	Coverage gaps and recovery time
Scalability	Limit communication and computation as nodes are added	Delay, load, and control error

10.1.1 Background and Tasks of Swarm Collaboration: Research Development

Swarm Research: Local Rules and System Integration

Period	Main Questions	Foundations Established
1990s to Early 2000s	How does collective behavior emerge? How should local rules be designed?	Concepts, bio-inspired algorithms, and theoretical models
Mid-2000s to 2010s	How can platforms support scale and heterogeneous collaboration?	Multi-vehicle experiments and application validation
Since the 2010s	How can learning, miniaturization, and human–swarm collaboration be integrated?	Transfer, robustness, and resilience assessment

Still ask: Are models suitable, information timely, and constraints satisfied throughout?

10.1.1 Background and Tasks of Swarm Collaboration: Coordination Research

Coordination Toward a Shared Goal

Research Area	Inspection Question	Observable Quantities
Distributed Control and Estimation	How can local information support a shared estimate?	State differences and estimation error
Formation and Motion Coordination	How can spacing be maintained through a turn?	Formation error and minimum distance
Task and Behavior Coordination	Who covers each area, and when is it handed over?	Coverage, conflicts, and time
Communication and Shared Perception	How can blind spots and stale data be addressed?	Age of information and visible area
Robustness and Adaptation	How can collaboration recover after a node leaves?	Reconfiguration time and remaining capability

10.1.1 Background and Tasks of Swarm Collaboration: Strategic Interaction

Game Models Include Other Decision Makers

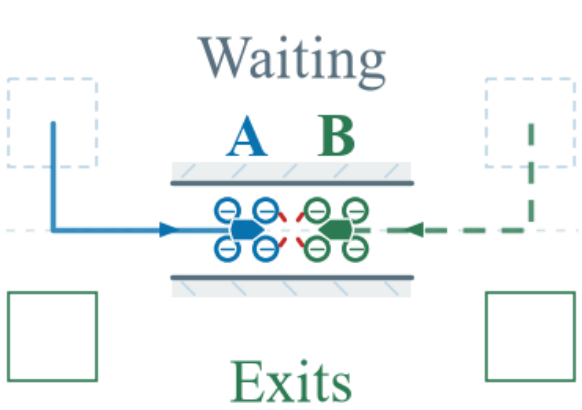
Research Area	Question to Answer
Game Modeling	How are players, information, actions, and payoffs defined?
Opponent Modeling and Intent Inference	What might the other player do, and how reliable is the inference?
Strategic Planning	How should actions and priorities be chosen with limited resources?
Reinforcement Learning and Policy Evolution	How can interaction data improve long-term decisions?
Perception and Decision Integration	How should feedback guide actions under incomplete observations?

10.1.1 Background and Tasks of Swarm Collaboration: Joint Choices

One Action, Different Joint Outcomes

Both go

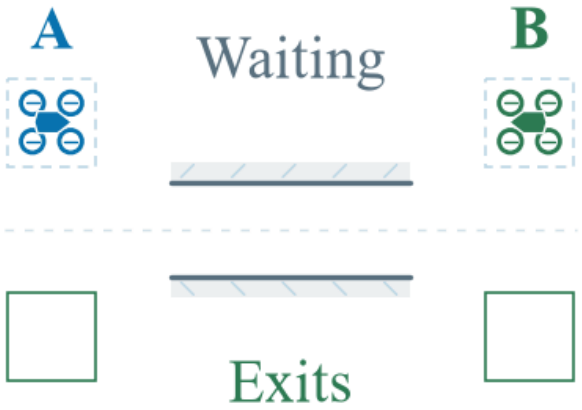
Conflict: neither finishes



Both stop in the corridor

Both wait

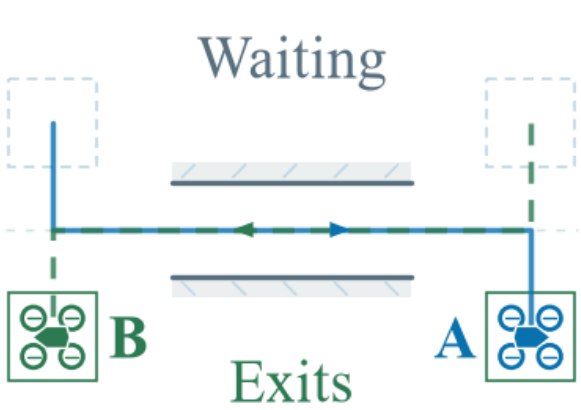
No progress



Both keep waiting

Agreed: B then A

Both pass



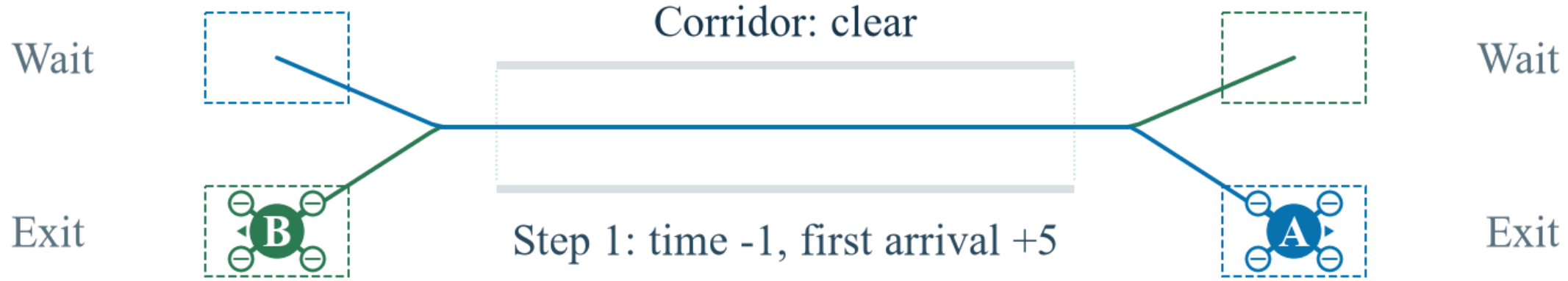
Both reach the exits

Joint choices by A and B determine safe passage through the same narrow corridor

10.1.1 Background and Tasks of Swarm Collaboration: Interaction Experience

Experience Recorded After an Action

Two-step recap: A waits -1, then crosses +4



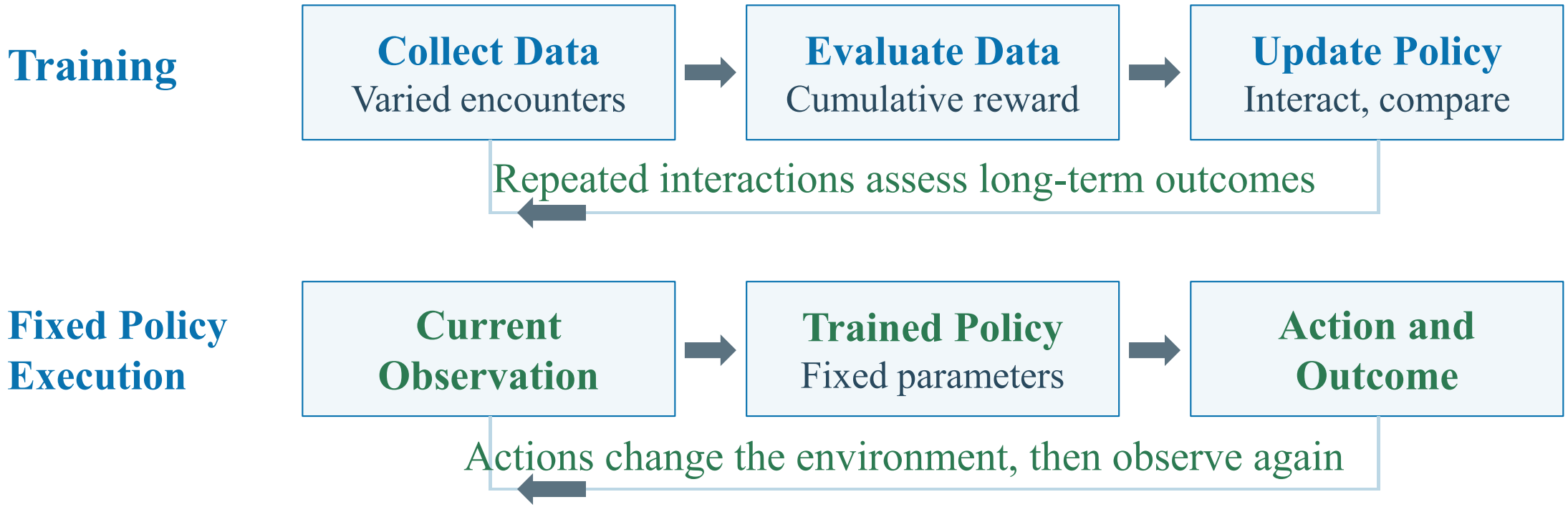
Local observation	Action taken	Reward	Next observation	Episode ended
Clear; B exited	Cross	+4	Clear; B exited	**true**
A not at goal	Step 1	-1 + 5	A at goal	**A is done**

Record A's observation, action, reward, next observation, and whether the task terminates

10.1.1 Background and Tasks of Swarm Collaboration: Training and Execution

Feedback and Policy Learning

Policy: a rule for choosing an action from current observations and, when needed, history.



The same policy parameters can choose waiting or passage according to the current situation.

10.1.1 Background and Tasks of Swarm Collaboration: Available Information

Centralized Training With Decentralized Execution (CTDE)

Centralized Training

A and B: states and actions
Both rewards, next states, and termination flags

Joint evaluation improves each policy

Global data: evaluation and updates.
Policies use inputs available at runtime.

Decentralized Execution

Own observations: position, corridor occupancy
Neighbor messages: B's position, time, and validity

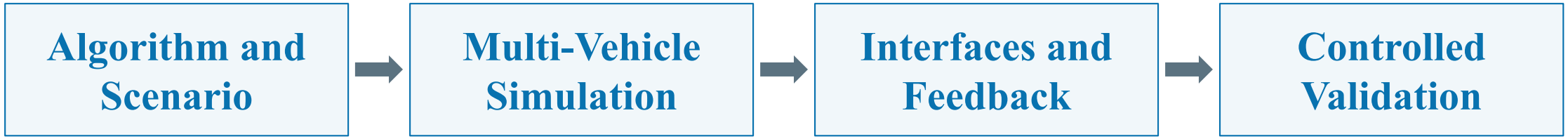
A's policy: wait or pass

Handle missing/stale messages as designed.
Do not use unavailable global truth.

Environment: interactions; training: update/save policies; execution: load/use policies.

10.1.1 Background and Tasks of Swarm Collaboration: Development Tools

Repeatable Validation Through Simulation



Repeatability: fix initial states, tasks, and random disturbances to compare methods.

Add realism in stages: motion model → flight-controller tracking → network and perception errors → physical constraints.

Assess task completion, formation error, safety distance, and communication timeliness together.

Chapter Outline

Swarm Collaboration Fundamentals and Control Architectures

Chapter 10 Swarm Collaboration and Game-Theoretic Confrontation

10.1 Swarm Fundamentals and Architectures

10.2 Consensus and Formation Control

10.3 Game Theory and RL Fundamentals

10.4 Single-Agent RL Algorithms

10.5 Multi-Agent RL and Scaling

10.6 Technologies and Challenges

10.7 RflySim Framework and Interfaces

10.8 Experiments and Validation

10.9 Exercises and Classroom Discussion

3 Units in This Section

10.1.1 Background and Tasks of Swarm Collaboration

Textbook: Sec. 10.1.1

10.1.2 Swarm Control Architecture

Textbook: Sec. 10.1.2.1

10.1.3 Decomposition of Swarm Control Tasks

Textbook: Sec. 10.1.2.2

10.1.2 Swarm Control Architecture: Overview

Control Architectures: Information and Command Flows

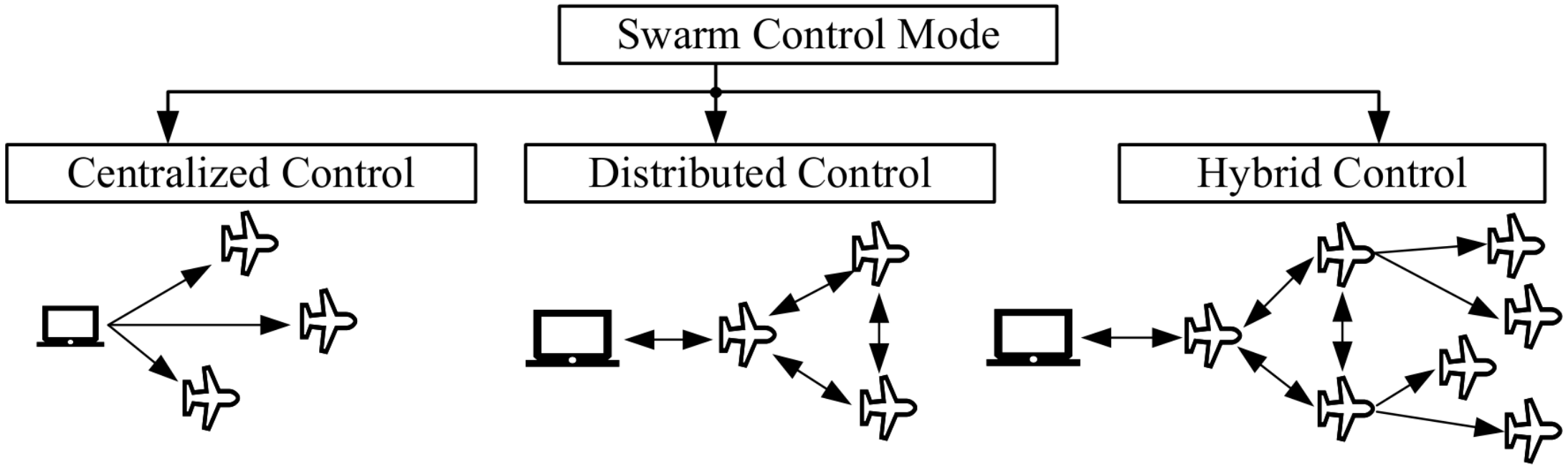


Fig. 10.2 Swarm System Control Architecture

Trace who observes states, computes coordination commands, and tracks and executes them.

10.1.2 Swarm Control Architecture: Centralized Information Flow

Centralized Control: Coordination at the Center

A, B, C: position, velocity

Goal + A, B, C states

One command per vehicle

Onboard Sensing and Estimation → **Central Coordination Computation** → **Onboard Tracking and Execution**

Center inputs: the global task, received vehicle states, and constraints.

Local responsibilities: state estimation, fast tracking control, actuation, and health feedback.

Global information aids joint optimization but centralizes communication and computation.

10.1.2 Swarm Control Architecture: Distributed Information Flow

Distributed Control: Local Coordination Computation

A goal may be predefined. Each update uses neighbor states, timestamps, and validity flags.



Neighbor: B

Neighbors: A, C

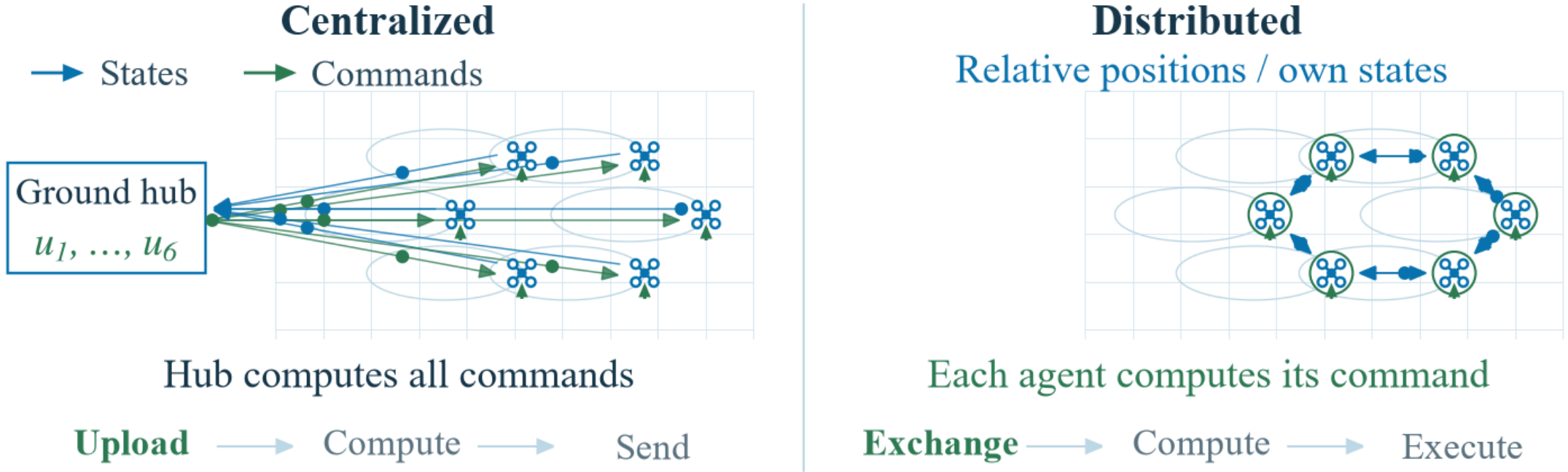
Neighbor: B

If B's message is stale, A cannot treat it as current. Predict, hold, or degrade as designed, then recheck the control assumptions.

10.1.2 Swarm Control Architecture: Same-Task Comparison

Centralized and Distributed Formation Computation

Same six initial states and formation goal, with a predefined shared reference trajectory.



Same task, different computation organization. Each vehicle still senses and executes locally.

10.1.2 Swarm Control Architecture: Same-Law Comparison

The Same Control Law at Different Computation Sites

Dimensionless states are 0, 4, and 8. Each node sums neighbor state minus own state.

Node	Data Needed for This Update	Sum of Neighbor Differences
A	Own state: 0; B: 4	$4 - 0 = 4$
B	Own state: 4; A: 0; C: 8	$(0 - 4) + (8 - 4) = 0$
C	Own state: 8; B: 4	$4 - 8 = -4$

Centralized: the center uses all states to produce three outputs.

Distributed: each node produces its own output and continually exchanges neighbor states.

10.1.2 Swarm Control Architecture: Hybrid Architecture

Hybrid Control: Global Roles and Local Feedback

Global Scheduling: Areas, Roles, Constraints

Assign tasks to subgroup A–B

Assign tasks to member C

Local Coordination: Neighbor States and Formation

Local Coordination: Neighbor States and Formation

Commands down; states up

Commands down; states up

Local Feedback: Tracking and Execution

Local Feedback: Tracking and Execution

Execution status and faults feed up the layers; reassign tasks when needed.

10.1.2 Swarm Control Architecture: Three Dimensions

Architecture, Deployment, and Model Fidelity

Dimension	Question to Ask	Examples
Control Architecture	Who computes coordination commands, from which information?	Centralized, distributed, hybrid
Computation Deployment	Which computing devices run the programs?	Multiple processes on one computer; several computers
Model Fidelity	What model generates each vehicle's motion?	Dynamics with flight-controller feedback; point-mass model

Identify required control information before checking where programs run.

10.1.2 Swarm Control Architecture: Trade-Offs and Limits

Architecture Trade-Offs: Information, Load, and Failure

Organization	Main Benefit	Constraints to Address
Centralized	Unified scheduling of global tasks	Center failure, aggregation bandwidth, and delay
Distributed	Local communication and autonomous computation	Connectivity, convergence speed, and incomplete information
Hybrid	Global task allocation with local response	Layer interfaces, conflict arbitration, and recovery
Independent Preset Trajectories	Execute predefined tasks on a shared schedule	Deviations, unexpected obstacles, and online replanning

Chapter Outline

Swarm Collaboration Fundamentals and Control Architectures

Chapter 10 Swarm Collaboration and Game-Theoretic Confrontation

10.1 Swarm Fundamentals and Architectures

10.2 Consensus and Formation Control

10.3 Game Theory and RL Fundamentals

10.4 Single-Agent RL Algorithms

10.5 Multi-Agent RL and Scaling

10.6 Technologies and Challenges

10.7 RflySim Framework and Interfaces

10.8 Experiments and Validation

10.9 Exercises and Classroom Discussion

3 Units in This Section

10.1.1 Background and Tasks of Swarm Collaboration

Textbook: Sec. 10.1.1

10.1.2 Swarm Control Architecture

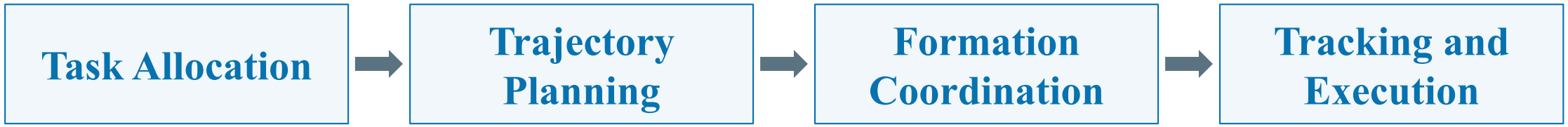
Textbook: Sec. 10.1.2.1

10.1.3 Decomposition of Swarm Control Tasks

Textbook: Sec. 10.1.2.2

10.1.3 Decomposition of Swarm Control Tasks: Tasks to Commands

Four Layers Connect Shared Tasks to Vehicle Actions

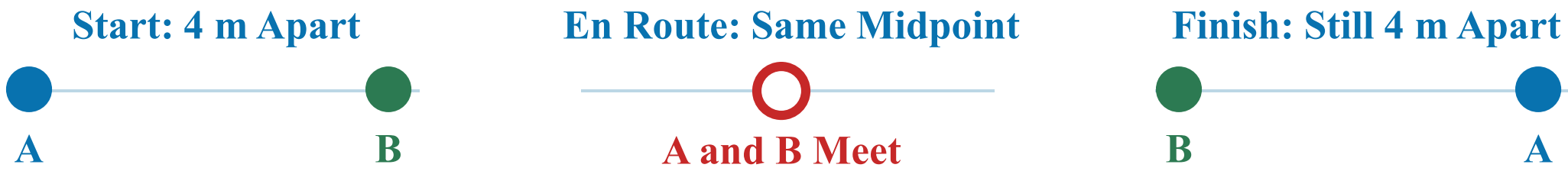


Layer	Input	Output
Task Allocation	Areas, capabilities, remaining resources	Tasks and priorities for each vehicle
Trajectory Planning	Goals, obstacles, motion constraints	Position or velocity references with timing
Formation Coordination	References, own state, neighbor states	Adjusted motion commands for each vehicle
Tracking and Execution	Motion command and own state	Control inputs and actual motion feedback

10.1.3 Decomposition of Swarm Control Tasks: Distinct Constraints

Trajectory, Formation, and Collision Constraints

Problem	Constrained Quantity	Three-Vehicle Inspection Criterion
Trajectory Planning	Each vehicle's reference motion over time	Feasible trajectories, reachable areas, no obstacle intersection
Formation Keeping	Relative geometry and velocity between members	Small relative-position errors and coordinated velocities
Collision Avoidance	Vehicle and obstacle distances throughout motion	Minimum distance above the specified safety limit



Swapping targets on one line can cause a collision despite safe final spacing.

10.1.3 Decomposition of Swarm Control Tasks: Simplified Calculation

Fixed-Target Example: Position Error to Velocity

Assumptions: ideal 1D velocity response, no disturbance or delay, fixed targets, gain 0.5 s^{-1}. Target spacing: 2 m. Recompute and hold velocity every 0.5 s. Right is positive.

i: node; p_i: position; p_i^{des}: target; v_i^*: velocity command; p_i^+: next position.

$$v_i^* = 0.5(p_i^{\mathrm{des}} - p_i), \qquad p_i^+ = p_i + 0.5v_i^*$$

Node	Position / m	Target / m	Velocity / (m/s)	After 0.5 s / m
A	0	2	1	0.5
B	4	4	0	4
C	8	6	−1	7.5

10.1.3 Decomposition of Swarm Control Tasks: Fixed-Target Response

Convergence and Safety Require Separate Checks

$$p_i[k] = p_i^{\mathrm{des}} + (p_i[0] - p_i^{\mathrm{des}})(0.75)^k, \quad t_k = 0.5k\,\mathrm{s}$$

k: update count; $p_i[0]$: initial position; t_k: elapsed time.

Time / s	A Position / m	B Position / m	C Position / m	Adjacent Gap / m
0	0	4	8	4
2	1.367	4	6.633	2.633
4	1.800	4	6.200	2.200
10	1.994	4	6.006	2.006

Here, adjacent distances approach 2 m from 4 m and stay at or above 2 m throughout.

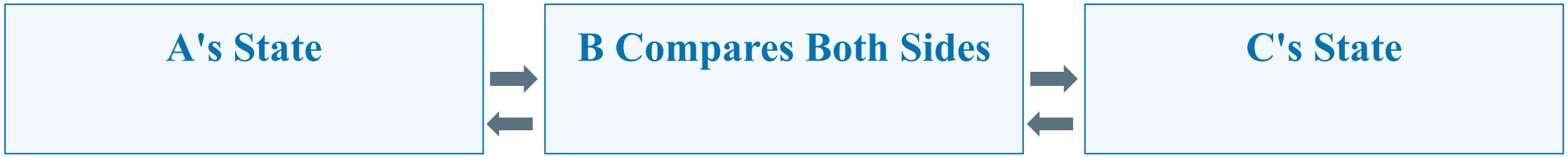
10.1.3 Decomposition of Swarm Control Tasks: Assessment Criteria

Multiple Metrics for the Same Task

Assessment	What to Record	Supported Conclusion
Shared Task	Coverage, missed areas, completion time	Was the assigned task completed?
Motion and Formation	Position tracking and relative-offset errors	Was the reference tracked and the structure maintained?
Safety Throughout	Minimum vehicle and obstacle distances over the full run	Were specified separation constraints satisfied?
Information and Faults	Age of information, outages, recovery time	Did the result depend on ideal information?

10.1.3 Decomposition of Swarm Control Tasks: Toward Consensus

Neighbor Errors Propagate Through the Swarm



Information: valid neighbor data determine the difference terms in each update. Check receipt and freshness first.

Control: relative errors set command directions, and updated states feed back into the next calculation.

Formation: coordinate states after subtracting desired offsets to retain spacing between members.

Exercises and Discussion

Discussion: Assessing Three-Vehicle Collaboration

Case 1: Three computers simulate three vehicles. One controller reads all states and sends all commands. Identify the architecture and explain why.

Case 2: Vehicles follow synchronized preset trajectories without exchanging neighbor states. Which capabilities remain unproven?

Case 3: Final formation error is small. Name three other categories of results that must be recorded.

Case 4: Fixed policy parameters still produce different wait/pass actions from observations. Is this closed-loop?
If training used B's global ground truth, what must execution check?

Reference Answers

Answers: Information Dependencies and Task Evidence

Case 1: Centralized control uses global states to compute coordination commands. Multiple computers describe deployment.

Case 2: This demonstrates synchronized preset execution. Online neighbor coordination, reconfiguration, and avoidance need separate evidence.

Case 3: Record coverage and time, minimum distance over the full run, data age and fault recovery. Also inspect the error history.

Case 4: It remains closed-loop because observations continually affect actions, with no online parameter update.
Keep policy-input semantics consistent between training and execution. Use extra global data only for permitted evaluation and updates.

Section Summary

Shared Tasks and Computable Feedback Loops

Concepts: a multi-agent system has multiple decision makers. Collaboration coordinates their work toward a shared goal.

Architecture: distinguish centralized, distributed, and hybrid control by information dependencies and command organization.

Task layers: task allocation → trajectory planning → formation coordination → local tracking.

Common errors: computer counts, final patterns, and algorithm names cannot replace convergence and safety evidence.

After choosing an architecture, express neighbor errors as a control law with explicit conditions.