Chapter Outline

Chapter Experiments and Integrated Validation

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

4 Units in This Section

10.8.1 Experiment Entries and Basic Validation

Textbook: Sec. 10.3.1

10.8.2 Formation and Heterogeneous Following

Textbook: Sec. 10.3.1, advanced experiments

10.8.3 Advanced Cases and Implementation Scope

Textbook: Sec. 10.3.2

10.8.4 Quantitative Assessment and Integrated Validation

Textbook: Secs. 10.3.1–10.3.2, integrated use

10.8.1 Experiment Entries and Basic Validation: Three Experiment Groups

Choose Experiments by Their Learning Goal

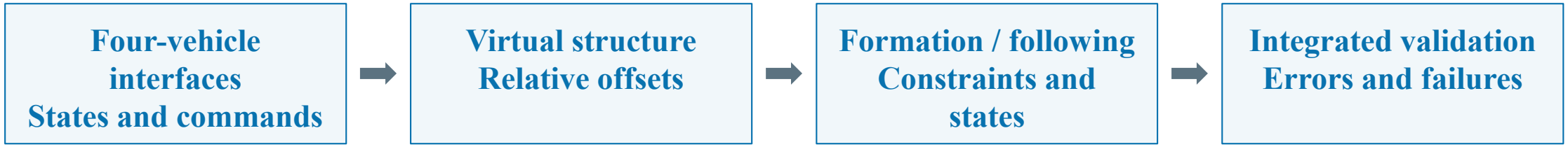
Root: [installation directory]/RflySimAPIs/10.RflySimSwarm/
Index: Index.pdf; interfaces: API.pdf; read each experiment's Readme.pdf first.

Directory	Learning Task	Representative Content
0.ApiExps	Identity, communication and control interfaces	Python / Simulink, HITL and terrain height
1.BasicExps	Basic motion, perception and feedback	Point-mass and fixed-wing models, collisions and lights
2.AdvExps	Coordination and engineering extensions	Virtual structures, formation changes, heterogeneity and multiple hosts

3.CustExps requires the Full edition or higher; also check dependencies and resources.

10.8.1 Experiment Entries and Basic Validation: From Interfaces to Tasks

A Stepwise Route for Locating Experiment Problems



First check: object IDs, local and common frames, control modes and continuing state updates.

Then compare: desired and actual trajectories, formation error, minimum separation and velocity limits.

Finally add: network delay, perception error and node failures, changing one condition at a time.

10.8.1 Experiment Entries and Basic Validation: Similar Targets, Different Loops

Four-Vehicle MAVLink and Point-Mass Examples

Common parent: 0.ApiExps/e2.PyRflySwarmAPIExps/

Model	Subdirectory and Entry	Initialization and Cleanup
PX4 / MAVLink	1.MAVLinkFull4Swarm/ MAVLinkFull4Swarm.bat / .py	initOffboard requests arming and writes failsafe parameters. Check landing and disarming before cleanup.
Python point mass	2.QuadGMAPIExp/ NoPX4SITL4Swarm.bat / .py	initPointMassModel; EndPointMassModel

If still armed, stopRun requests forced disarming; it does more than stop reception.

10.8.1 Experiment Entries and Basic Validation: Prepare, Run, Finish

Starting and Finishing the Four-Vehicle MAVLink Example

Stage	Documented Operation	Required Feedback
Prepare	Save and end other simulations. Check IDs, frames and parameter writes.	BAT needs administrator rights. It ends CopterSim; cleanup closes ground-station and 3D tools.
Start scene	Run MAVLinkFull4Swarm.bat. Continue after all four vehicles are ready.	GPS 3D fixed & EKF initialization finished
Run control	In the Python38Run.bat terminal: python MAVLinkFull4Swarm.py	Check mode, position and altitude. S / T show IDs / trajectories in 3D.
Finish	Default: fixed waits. Improved finish: verify landing and disarming before cleanup.	stopRun may force disarming. It is not a landing-state check.

10.8.1 Experiment Entries and Basic Validation: A Worked Boundary Check

A Descent Command Does Not Establish Landing

Assume level ground at local z = 0 and exact downward-velocity tracking. Start at z = −3 m; descend for 10 s.

$$\Delta z = v_z T = 0.2 \times 10 = 2 \text{ m}$$

$$z_{\text{end}} = -3 + 2 = -1 \text{ m}$$

The vehicle remains 1 m above ground. Cleanup follows a fixed wait; stopRun may force disarming if still armed. Verify landing and disarming first.

10.8.1 Experiment Entries and Basic Validation: Interpreting Evidence

What Each Basic Experiment Validates

Topic	Input and Output	Supported Check
Terrain / fixed-wing point mass	Map height and waypoints to initial state and route motion	Frames, terrain initialization and waypoint changes
Visual following	Red-target center / size error to velocity	Proportional visual servoing; zero velocity on target loss
Collision detection	Opposing trajectories to collision events and IDs	The detection-feedback path, not collision avoidance
Lights / figure-eight motion	Display settings / reference trajectories to scene objects	Display or tracking; separately check dynamics and separation

10.8.1 Experiment Entries and Basic Validation: Simulated Detection

Associate Collision Events with Trajectory Records

Directory: 1.BasicExps/e5.CollisionExps/2.CollMAVLinkAPI_Py/
Entry: CollisionDemo.bat then CollisionDemo.py

Opposing vehicle paths → **3D collision detection** → **isVehicleCrashID** → **Align events and trajectories**

Enable the documented collision feedback, check both IDs and run the opposing motion in simulation.

Match trajectories, collision IDs and event states in time. Distinguish no collision, no detection and no returned feedback.

Chapter Outline

Chapter Experiments and Integrated Validation

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

4 Units in This Section

10.8.1 Experiment Entries and Basic Validation

Textbook: Sec. 10.3.1

10.8.2 Formation and Heterogeneous Following

Textbook: Sec. 10.3.1, advanced experiments

10.8.3 Advanced Cases and Implementation Scope

Textbook: Sec. 10.3.2

10.8.4 Quantitative Assessment and Integrated Validation

Textbook: Secs. 10.3.1–10.3.2, integrated use

10.8.2 Formation and Heterogeneous Following: Control Principle

Virtual Structure: A Common Center and Individual Offsets

Center trajectory Changes with time	+	Fixed offsets Define the formation	=	Position targets Tracking controllers

$$p_i^*(t) = c(t) + d_i, \qquad p_i^* - p_j^* = d_i - d_j$$

Targets, center and offsets use meters in one frame. Fixed offsets retain desired geometry, not guaranteed actual separation or collision freedom.

10.8.2 Formation and Heterogeneous Following: Center Plus Offset

One Target Calculation from the Current Example

$$c(t) = \begin{bmatrix} 0.75\cos(0.15t) \\ 0.5\sin(0.3t) \end{bmatrix} \text{ m}$$

Time: s; angular frequency: rad/s. Substitute t = 0 using the first three triangular offsets.

Vehicle	Offset / m	Center / m	Target / m
1	(0,1)	(0.75,0)	(0.75,1)
2	(1,0)	(0.75,0)	(1.75,0)
3	(-1,0)	(0.75,0)	(-0.25,0)

Target 2 minus target 3 is (2, 0) m: an analytical target calculation, not a measured result.

10.8.2 Formation and Heterogeneous Following: Documented Interface

Mode and Trajectory Choices in the Virtual-Structure Example

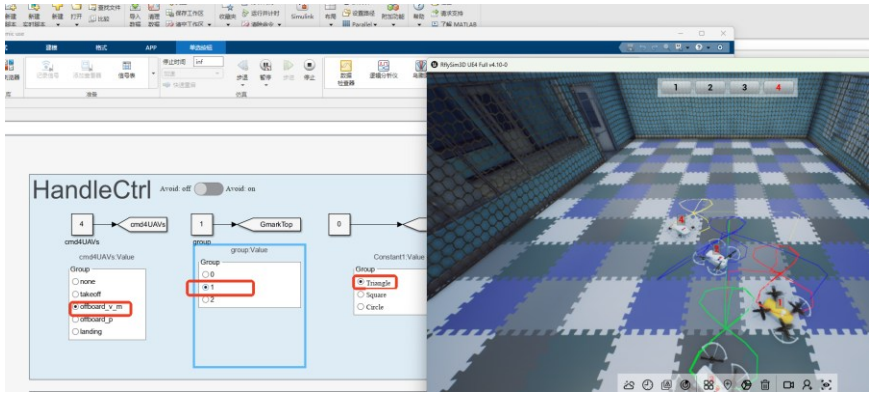


Figure: documented controls and scene

Stage	Settings and Actions for Six-Vehicle SITL
Setup	UDPmode=Simpledata; 127.0.0.1. Initialize six vehicles; verify EKF readiness.
Mode	none, then run, then takeoff. Once stable: offboard_v_m or offboard_p.
Trajectory	Center switch0 /1 /2: hold / figure eight / circle. The rightmost control selects formation.
Finish	Select land at left. Confirm landing and disarming, then stop the controller.

2.AdvExps/e1.MatCentCtrlUAVsSim/7.RflyUdpSimpleVirStruct/
iniScript4Model.m → SITLRun6uavs.bat → multiuav.slx

10.8.2 Formation and Heterogeneous Following: Result Metrics

Tracking Error and Formation Error Are Different Metrics

$$e_i(t) = p_i(t) - p_i^*(t)$$

$$e_{ij}^F(t) = (p_i - p_j) - (d_i - d_j) = e_i - e_j$$

Use a common frame and matched timestamps. Report individual error, relative formation error and minimum separation over the full interval.

If every vehicle is 0.5 m east of its target, relative geometry may be correct while absolute tracking is wrong.

10.8.2 Formation and Heterogeneous Following: Documented Result

Targets and Transients in a Three-Vehicle Formation Change



Figure: documented three-vehicle horizontal-line formation

Targets: switch among triangle, horizontal line and vertical line.

Structure: target attraction, inter-vehicle repulsion and velocity limits.

Record trajectories, formation error, minimum separation and settling time.

Directory: 2.AdvExps/e5.AntiCollFormCtrlExps/3.SwarmEarthVelCtrlCollUdp_Mat/

10.8.2 Formation and Heterogeneous Following: Transient Validation

Quantitative Checks for a Three-Vehicle Formation Change

Check initial states and settings → **Run model Controller trigger: 1** → Timestamp the switch → Measure the full transition

Entry: SwarmFormSITL.bat then SwarmFormEarthVelCTRLUdpAPI.slx
Parameters: Init_control.m

Illustrative Criterion	Measurement	Interpretation
Center separation at least 1.2 m	Check every pair from before the switch to the end	Also account for body size, localization error and sampling
Error ≤ 0.2 m by 10 s; hold for 2 s	Compare desired offsets for each pair in the common frame	Distinguish briefly crossing a threshold from a settled formation

10.8.2 Formation and Heterogeneous Following: Air–Ground Cooperation

Following a Rover: Target State Becomes a Position Command

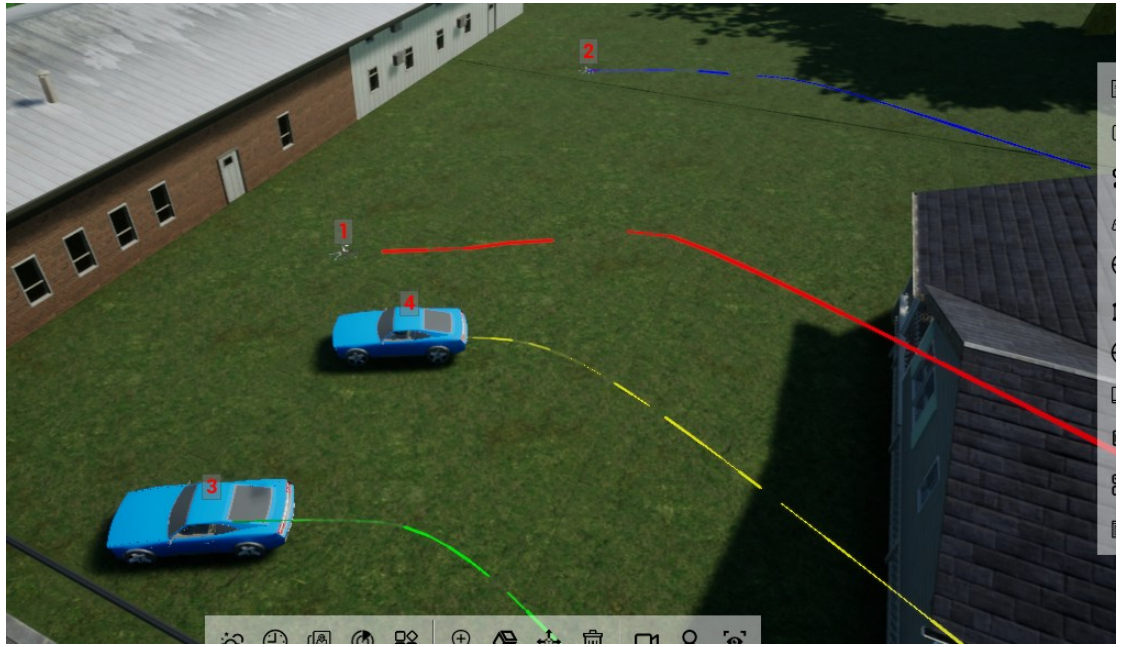


Figure: documented air–ground state-following example

Sequence: the UAV reaches its initial altitude before following.

Target: use the rover's horizontal position while holding altitude.

Check horizontal error, altitude error, turn lag and response to target loss.

2.AdvExps/e6.Hetero_SwarmExps/2.Car_UAVCoTraking/
MultiVehicleRunALL.bat → Python38Run.bat → UDPSimpleMultiVehicleSwarm.py

10.8.2 Formation and Heterogeneous Following: Deployment and Models

Keep Comparable Task Metrics When Extending the System

Extension	Representative Path / File	Retained and Added Checks
Eight vehicles on two PCs	2.AdvExps/e2.MatDISCtrlUAVsSim/ 1.BroadNetSwarm_Mat/	Keep trajectory-error definitions. Add state age and update overruns.
PX4 plus point mass	2.AdvExps/e4_PX4+MPSwarm/ HybridSwarmCtrl.py	Retain relative error; compare model dynamics
Air–ground model configuration	CarR1Diff.dll、MulticopterNoCtrl.dll、 Standard_Car_Blue.xml	Keep task criteria. Verify installed assets and the XML-selected models.

Back up the shared XML before replacement; restore it afterward to preserve other examples.

Chapter Outline

Chapter Experiments and Integrated Validation

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

4 Units in This Section

10.8.1 Experiment Entries and Basic Validation

Textbook: Sec. 10.3.1

10.8.2 Formation and Heterogeneous Following

Textbook: Sec. 10.3.1, advanced experiments

10.8.3 Advanced Cases and Implementation Scope

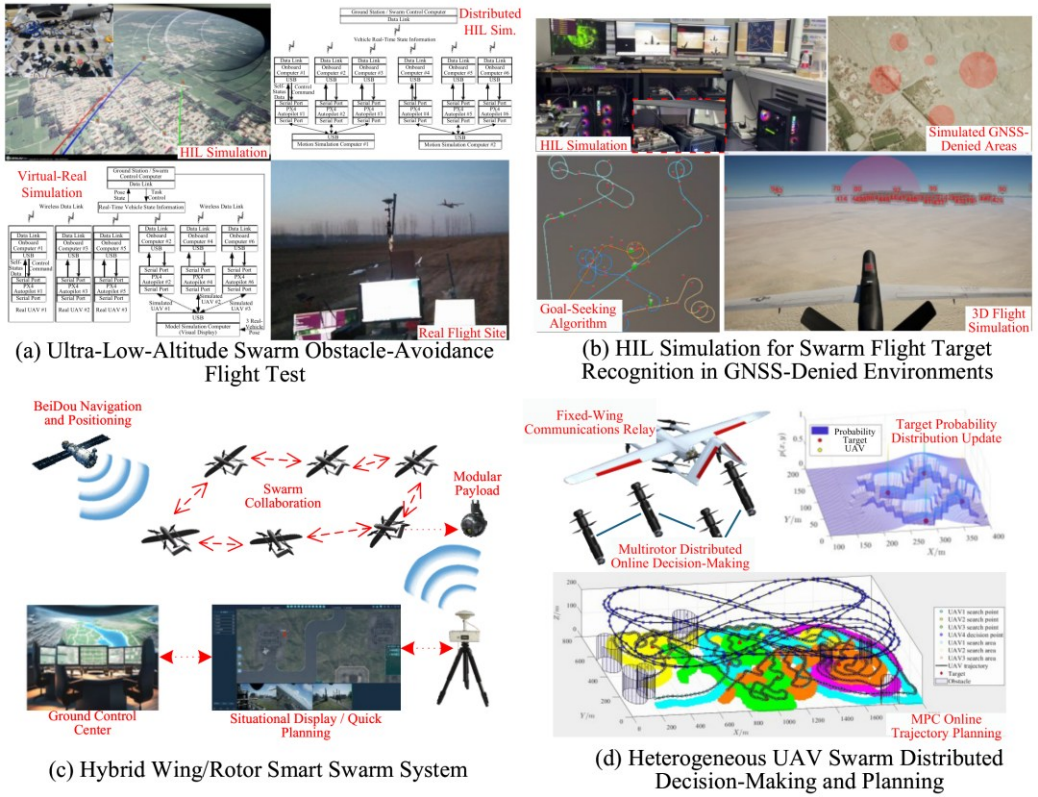
Textbook: Sec. 10.3.2

10.8.4 Quantitative Assessment and Integrated Validation

Textbook: Secs. 10.3.1–10.3.2, integrated use

10.8.3 Advanced Cases and Implementation Scope: Tasks and Methods

Four Advanced Cases in the Textbook



(a) Ultra-Low-Altitude Swarm Obstacle-Avoidance Flight Test

(b) HIL Simulation for Swarm Flight Target Recognition in GNSS-Denied Environments

(c) Hybrid Wing/Rotor Smart Swarm System

(d) Heterogeneous UAV Swarm Distributed Decision-Making and Planning

Figure 10.10. Overview of advanced cases

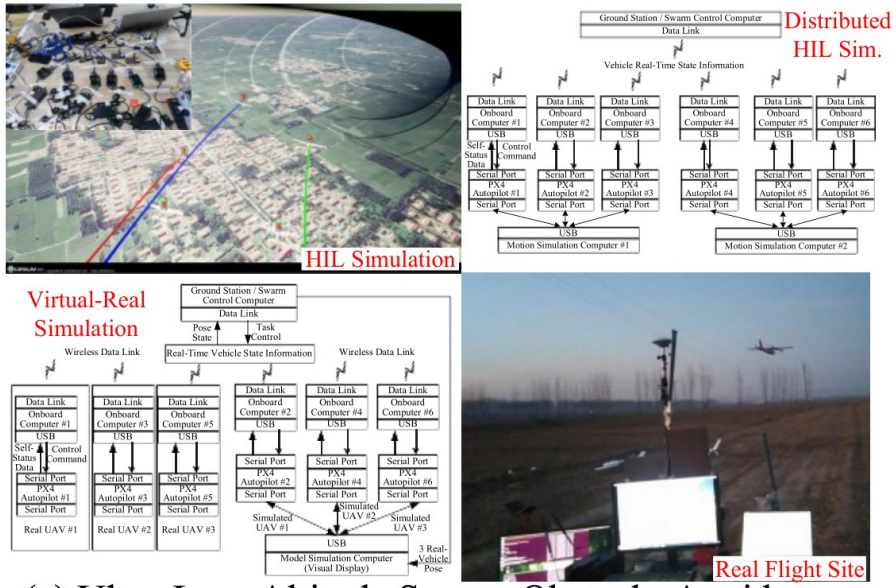
Match each case to its task, observations and decision outputs.

Trace perception, communication and control on one timeline.

Pictures show configurations. Use records to assess performance.

10.8.3 Advanced Cases and Implementation Scope: Position, Time, Prediction

Ultra-Low-Altitude Swarm Avoidance



(a) Ultra-Low-Altitude Swarm Obstacle-Avoidance Flight Test

Figure 10.10(a). Low-altitude swarm avoidance

Network Time Protocol (NTP) + fused pose → packet-loss compensation → predictive control.

Measure ground and obstacle clearance, vehicle separation, state age and control period.

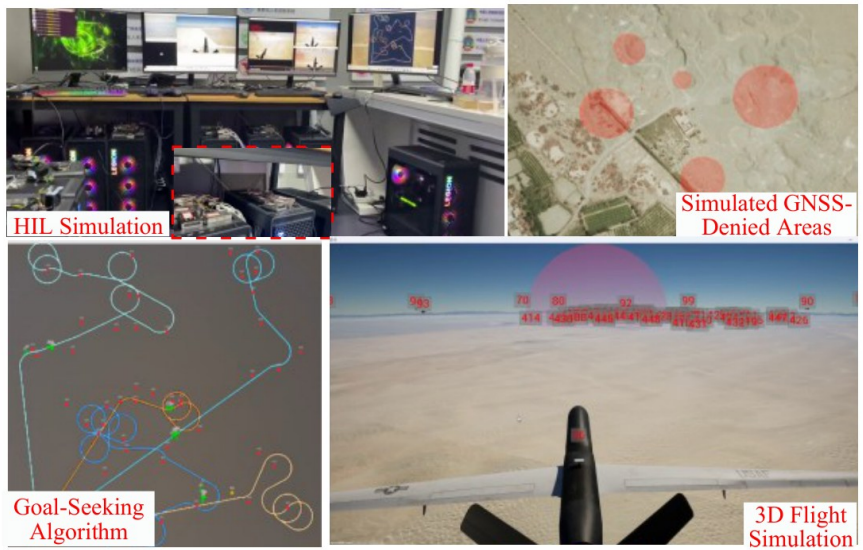
Fix reference paths and initial states. Vary clock error, packet loss and compensation separately.

Illustrative criteria: obstacle clearance≥0.5 m for 60 s; clock offset≤ 5 ms.

10.8 Experiments and Validation

10.8.3 Advanced Cases and Implementation Scope: Visual Navigation

Swarm Flight Without Global Navigation Satellite Systems (GNSS)



(b) HIL Simulation for Swarm Flight Target Recognition in GNSS-Denied Environments

Figure 10.10(b). Flight and recognition without GNSS

Images / point clouds support visual localization, map relocalization and recognition to generate flight targets.

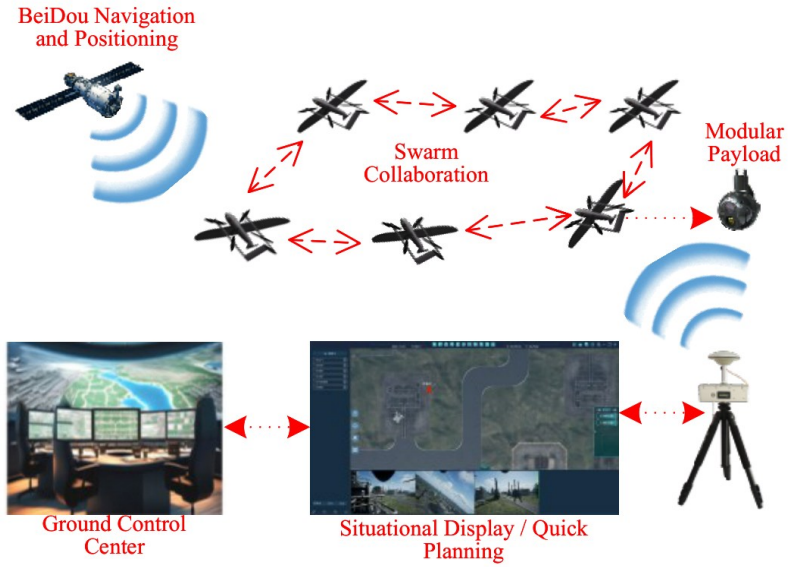
Measure drift, relocalization time, recognition accuracy and task completion.

Vary texture, lighting and occlusion. Check hold, exit and recovery responses.

Illustrative criteria: localization RMSE≤0.3 m and recognition F1≥0.90, using reference truth. RMSE: root mean square error. F1: harmonic mean of precision and recall.

10.8.3 Advanced Cases and Implementation Scope: Heterogeneous Roles

A Hybrid Fixed-Wing and Multirotor Swarm



(c) Hybrid Wing/Rotor Smart Swarm System

Figure 10.10(c). Fixed-wing / multirotor cooperation

Assign tasks, allocate roles and trajectories, then execute on each platform.

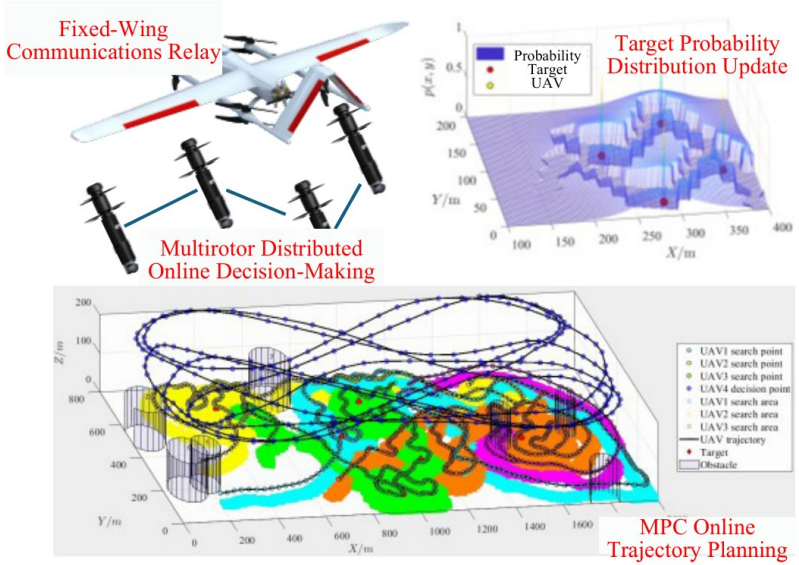
For each type, constrain speed, endurance, turns, payload and communication coverage.

Measure coverage, completion time, energy use and constraint violations separately.

Illustrative criteria: coverage ≥ 95% over the task window; no motion-limit violations.

10.8.3 Advanced Cases and Implementation Scope: Probability Maps and Replanning

Distributed Planning in a Heterogeneous Swarm



(d) Heterogeneous UAV Swarm Distributed Decision-Making and Planning

Figure 10.10(d). Distributed swarm planning

Update probability maps; use model predictive control (MPC); share results and replan.

Check prediction horizon, neighbor data, communication delay and path feasibility.

Measure coverage, information gain, repeated coverage and planning overruns.

Illustrative criteria: coverage ≥ 90%; at 5 Hz, planning-time P95 < 0.2 s.
P95: 95th percentile. Report overrun count, proportion and maximum time separately.

10.8.3 Advanced Cases and Implementation Scope: What Is Implemented

Distinguish Rule-Based Policies, Single-Agent PPO, and MARL

Material	Actual Content	Use in Study
MARLExps Custom material	Single-quadcopter proximal policy optimization (PPO); parallel environments	Study training-to-SITL interfaces; this is not a ready-made MADDPG example.
Area-defense example	Rule-based policy; dependencies vary with SDK version	Trace states and decisions. Check dependencies and scene before reproduction.
Textbook MADDPG / CTDE	Design training information and execution observations	Use Slides 10.5: define observations, actions, rewards, replay and joint assessment.

Custom entries require the Full edition or higher. Verify dependencies, weights and scenes.

Chapter Outline

Chapter Experiments and Integrated Validation

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

4 Units in This Section

10.8.1 Experiment Entries and Basic Validation

Textbook: Sec. 10.3.1

10.8.2 Formation and Heterogeneous Following

Textbook: Sec. 10.3.1, advanced experiments

10.8.3 Advanced Cases and Implementation Scope

Textbook: Sec. 10.3.2

10.8.4 Quantitative Assessment and Integrated Validation

Textbook: Secs. 10.3.1–10.3.2, integrated use

10.8.4 Quantitative Assessment and Integrated Validation: Assumptions and Parameters

An Analytical Three-Vehicle Tracking Example

$$\dot{p}_i = \dot{c} - k(p_i - c - d_i), \qquad e_i(t) = e_i(0)e^{-kt}$$

Assume first-order point dynamics, a common frame, no delay or noise. Use this section's center trajectory and offsets.

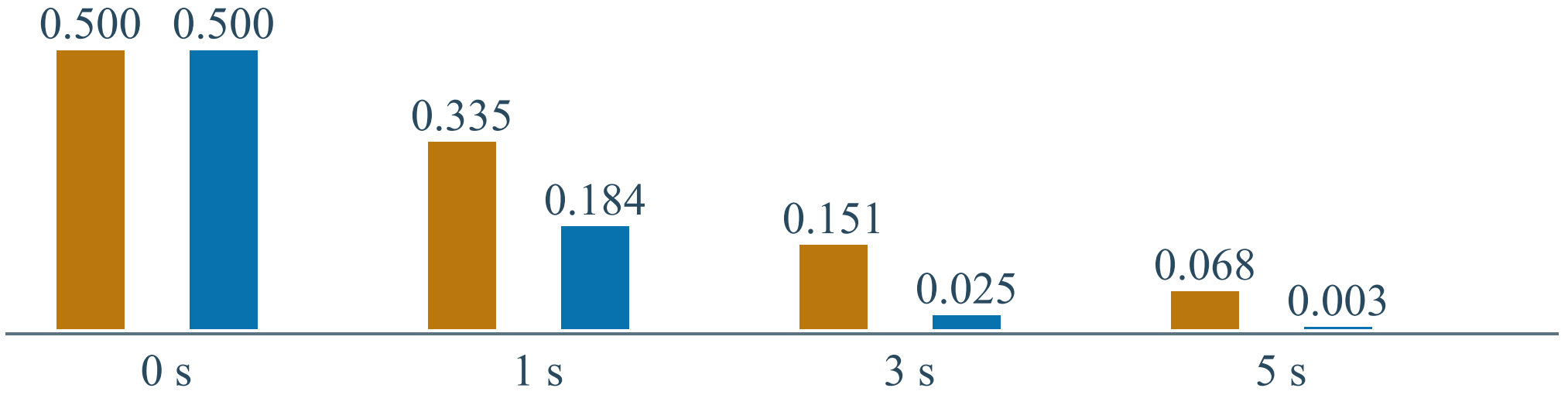
Teaching Setting	Value and Unit
Initial error / m	Vehicle 1: (0.4, 0.3); 2: (0.2, −0.2); 3: (−0.3, −0.1)
Gain and record	Gain: 0.4 or 1.0 /s. Compute analytical states every 0.01 s over 0–5 s.
Illustrative criteria	At 5 s: maximum error ≤ 0.05 m. Throughout 0–5 s: separation ≥ 1.2 m and speed ≤ 0.8 m/s.

10.8.4 Quantitative Assessment and Integrated Validation: Analytical Results

Gain Changes the Rate of Error Decay

Maximum individual error / m. Analytical model calculation, not an SDK run.

Orange: gain 0.4 /s. Blue: gain 1.0 /s.



At 5 s: 0.068 m fails the error criterion; 0.003 m passes.
Sampled minimum distances: 1.444 / 1.416 m. Maximum speeds: 0.244 / 0.427 m/s.

10.8.4 Quantitative Assessment and Integrated Validation: Criterion Limits

Sampled Separation and Continuous-Time Safety

$$d_{\min,continuous} \geq d_{\min,sampled} - 2v_{\max}\Delta t$$

$$1.4156 - 2 \times 0.8 \times 0.01 = 1.3996\ \text{m} > 1.2\ \text{m}$$

A conservative speed bound is center-speed bound plus gain times maximum initial error; it remains below 0.8 m/s here.

A point-model bound is not actual vehicle clearance. Include body size, localization error and stale states.

10.8.4 Quantitative Assessment and Integrated Validation: Controlled Comparisons

Design Comparisons for the Actual Examples

Stage	Fixed Conditions	Change and Compare
Interface baseline	IDs, common frame, initial states, targets and window	Establish command-to-feedback correspondence first
Model comparison	Reference, control level and statistics	Point-mass versus PX4 tracking and saturation
Network / perception	Reference trajectory, algorithm and control period	Vary delay, loss and localization noise separately
Failure and recovery	Task and trigger time	Responses to disconnection, invalid localization and departures

10.8.4 Quantitative Assessment and Integrated Validation: Logs and Windows

Minimum Records for Recalculating the Results

Record Group	Suggested Fields	Purpose
Identity and version	Run ID, script/model versions, vehicle IDs and settings	Reproduce conditions and detect crossed data
Time and mode	Generation, reception and use times; modes and validity	State age, deadline misses and failure branches
Task and motion	Common-frame position, target, velocity and command	Individual error, relative error and separation
Stages and events	Takeoff, switch, arrival, collision and finish times	Formation time, violation duration and completion rate

10.8.4 Quantitative Assessment and Integrated Validation: Interpretation

Locate Failures Along the Data Flow

Symptom	First Check	Distinguishing Evidence
The whole formation is shifted	Local/common origins and target generation	Relative geometry can be correct while absolute tracking is wrong
Large lag during turns	State age, velocity limits and control period	Align target, state and command in time
Separation drops during a change	Target assignment, transition paths and repulsion settings	Inspect the full transition, not just final geometry
Detection succeeds but task fails	Coordinate association, decision conditions and execution state	Align detection, target generation and mode feedback

Exercises and Discussion

Discussion: What Does the Evidence Establish?

Q1: Final separations exceed 2 m. Does this establish safety throughout the formation change? What record is missing?

Q2: Every vehicle is 0.5 m east of its target. What are the individual and relative formation errors?

Q3: Several single-agent PPO environments sample in parallel. Is this MADDPG? Explain using the data flow.

Reference Points

Reference Answers: Match Conclusions to Evidence

Claim	Reference Conclusion	Evidence Needed
1. Neat final geometry	Insufficient to establish safety over the whole transition	Pairwise paths, minimum separation, sampling / speed bounds and vehicle sizes
2. A common translation	Each tracking error is 0.5 m. Relative formation error is 0.	Compute absolute and relative errors in the common frame
3. Multiple PPO environments	Parallel sampling is not by itself cooperative MARL or MADDPG.	Joint task, agent observations/actions, and training/execution information flow

Section Summary

Tasks, Records, and Conclusions

Task: define inputs, constraints and success criteria before choosing interface, basic or advanced examples.

Relationships: a virtual center generates references; shared state supports following. Distinguish controller, deployment and model choices.

Evidence: separate textbook pictures, documented expectations, analytical calculations and measured records.

Assessment: record errors, separation, timing and failures on common frames and timelines; retain parameters and raw data.