Chapter Outline

Experiments and Simulation-to-Real Transfer

Chapter 8 Multimodal Perception and Intelligent Decision-Making

8.1 Image Processing Fundamentals

8.2 Recognition, Tracking, Optical Flow

8.3 Localization, Mapping, and Planning

8.4 Common Perception Sensors

8.5 Coordinate Transformations

8.6 Fusion and Control Decision-Making

8.7 Tools and Framework Interfaces

8.8 Experiments and Real-World Transfer

8.9 Exercises and Classroom Discussion

4 Units in This Section

8.8.1 Chapter Experiment List
Textbook: Sec. 8.3.1

8.8.2 Basic and Integrated Experiments
Textbook: Sec. 8.3.1 and Companion Examples

8.8.3 Advanced Case Studies
Textbook: Sec. 8.3.2

8.8.4 Simulation-to-Real Transfer
Textbook: Sec. 8.2.2

8.8.1 Chapter Experiment List: Categories

Use Index to Find a Task and Readme to Begin

[installation directory]/RflySimAPIs/8.RflySimVision/
Index.pdf; Readme.pdf in Each Experiment Folder

Folder	Teaching Role	Experiment Outputs
0.ApiExps	Interfaces and configuration	Valid images, point clouds, messages, and control requests
1.BasicExps	Basic use and task loops	Target responses, stage changes, task execution
2.AdvExps	Integrated sensing/planning; multiple vehicles	Maps, poses, trajectories, control results
3.CustExps	Custom advanced tasks	Behavior structures and multimodal task execution

8.8.1 Chapter Experiment List: Interface Learning

First Connect the Environment, Images, and Transport

[installation directory]/RflySimAPIs/8.RflySimVision/
0.ApiExps/; Readme.pdf in Each Folder Below

Actual Folder	Connection to Establish	Acceptance Check
0.Preparation	Runtime and platform processes	Required programs and dependencies available
1-UsageAPI	Configuration → first frame → continuous reception	Correct dimensions, units, and sample time
2-DistributedSimAPI	Observations/control across nodes	Match vehicle, IP address, port, and time base

Configuration, depth, and time: 0.ConfigJsonAPI, 3.DepthCameraDemo, 10.ReadTimeStmp.

8.8.1 Chapter Experiment List: Datasets and Point Clouds

One Scene Can Produce Different Algorithm Inputs

[installation directory]/RflySimAPIs/8.RflySimVision/
0.ApiExps/; Readme.pdf in Each Folder Below

Actual Folder	Input → Output	Learning Focus
3-VisionAIAPI	Scene/targets → images and labels	Capture, labels, and data organization
4.Point-CloudVisualize	Point cloud → 3D display	XYZ units, range, and axis directions
7.LidarLivoxDemo	Livox-style configuration → point cloud	Scan distribution, sampling, and timing
8.LidarAPIDemo	LiDAR parameters → spatial observations	Configuration, decoding, and algorithm inputs

8.8 Experiments and Real-World Transfer

8.8.1 Chapter Experiment List: Robot Operating System (ROS) and Gimbals

Publishing, Coordinate Trees, and Camera Control Have Distinct Roles

[installation directory]/RflySimAPIs/8.RflySimVision/
0.ApiExps/; Readme.pdf in Each Folder Below

Actual Folder	Role	Output to Check
5.VisCaptureMergeROSAPI	Package/publish sensor messages	encoding, stamp, and update rate
6.ConfigROSTFAPIDemo	Reference frames and transform (TF) tree	Frame names and mounting transforms
9.CameraInfo	Camera parameters and image interpretation	Intrinsics, distortion, field of view
12.UAV_PodSimUI	Camera and gimbal-pod operation	Gimbal angles, field of view, target readouts

8.8.1 Chapter Experiment List: Basic Use

Image Processing Leads to Target Responses

[installation directory]/RflySimAPIs/8.RflySimVision/
1.BasicExps/1-VisionCtrlDemos/; Readme.pdf in Each Folder Below

Actual Folder	Experiment Topic	Teaching Output
e1_CrossRingNoPX4	Ring passing	Target recognition, stage changes, motion response
e3_ShootBall	Moving-ball following	Target position over time and response
e6_Circle-follow	Visual Plate Following	Center, box height, pixel error, velocity
e7_ManDetect	Face recognition	Face boxes; no identity recognition

8.8.1 Chapter Experiment List: Integrated Systems

Assess Localization, Task Logic, and Control Together

[installation directory]/RflySimAPIs/8.RflySimVision/1.BasicExps/
Readme.pdf in each folder; SLAM = simultaneous localization and mapping.

Actual Folder	Task Level	Main Checks
2-BaseDemoAuto	Automatic task commands	Task order and execution state
3-BaseDemoManu	Interactive/manual tasks	Input actions and vehicle response
4_CompSlamNav	SLAM Indoor Competition	Map, pose, and task score
5.LLMUavComp	Language-Model Tasks	Task structure and feedback
6.LLMUavCompSLAM	Language + SLAM/Planning	Structure, pose, path, execution

8.8.1 Chapter Experiment List: Advanced Perception

Gimbals, SLAM, and Tracking Produce Different Outputs

2.AdvExps/; Readme.pdf. YOLO: You Only Look Once; KCF: Kernelized Correlation Filter.

Actual Folder	Textbook Topic	Outputs and Distinctions
e1_CameraKeyDemoOnUbuntu **e2_CameraKeyDemoOnWindows** **e3_CamerKeyROSDemo**	Gimbal control; image transport	Gimbal angles, field of view, images Ubuntu / Windows / ROS
e4_RflySimPlatform_SLAM **e6_LaserSLAMdemo**	Visual / LiDAR SLAM	Vehicle pose, trajectory, map Different sensor inputs
e7_ObjDetectYolo **e8_SingleObjTracking** **e9_Object-Follow**	Detection, tracking, visual servoing	e7: YOLO examples e8: KCF; ground-truth init e9: Ground-truth servoing

8.8.1 Chapter Experiment List: Planning and Tasks

Paths, Swarms, and Behavior Decisions Need Separate Checks

Planning/KCF/Swarm: 2.AdvExps/; Behavior/Language: 3.CustExps/.
KCF = Kernelized Correlation Filter

Actual Folder	Textbook Topic	Main Output
e10_Planner	A* path search	Feasible path through a map
e11_KCFdemo	KCF Ring Passing / Formation	Tracks; continuous task response
e13_EGOPlannerSwarm	EGO-Swarm planning	Timed, collision-free paths per vehicle
e7_BT_control **e10.LLM-BehaviorTreeUAVCtrl**	Behavior Trees / Language Tasks	Conditions, actions, and failure branches

8.8.1 Chapter Experiment List: Selection Example

Choose Three Routes by Input, Task, and Output

Given Task	Entry Point	Required Result
Red-Green-Blue (RGB) Red-Plate Following	e6_Circle-follow	Center, box height, velocity, motion response
LiDAR + Unknown Indoor Space	e6_LaserSLAMdemo	Map, continuous pose estimates, trajectory
Existing Map + Two Goals	e16_ESDFPathPlan	Current planning file and matching flight trajectory

Inputs select experiments; outputs define acceptance evidence.

Chapter Outline

Experiments and Simulation-to-Real Transfer

Chapter 8 Multimodal Perception and Intelligent Decision-Making

8.1 Image Processing Fundamentals

8.2 Recognition, Tracking, Optical Flow

8.3 Localization, Mapping, and Planning

8.4 Common Perception Sensors

8.5 Coordinate Transformations

8.6 Fusion and Control Decision-Making

8.7 Tools and Framework Interfaces

8.8 Experiments and Real-World Transfer

8.9 Exercises and Classroom Discussion

4 Units in This Section

8.8.1 Chapter Experiment List
Textbook: Sec. 8.3.1

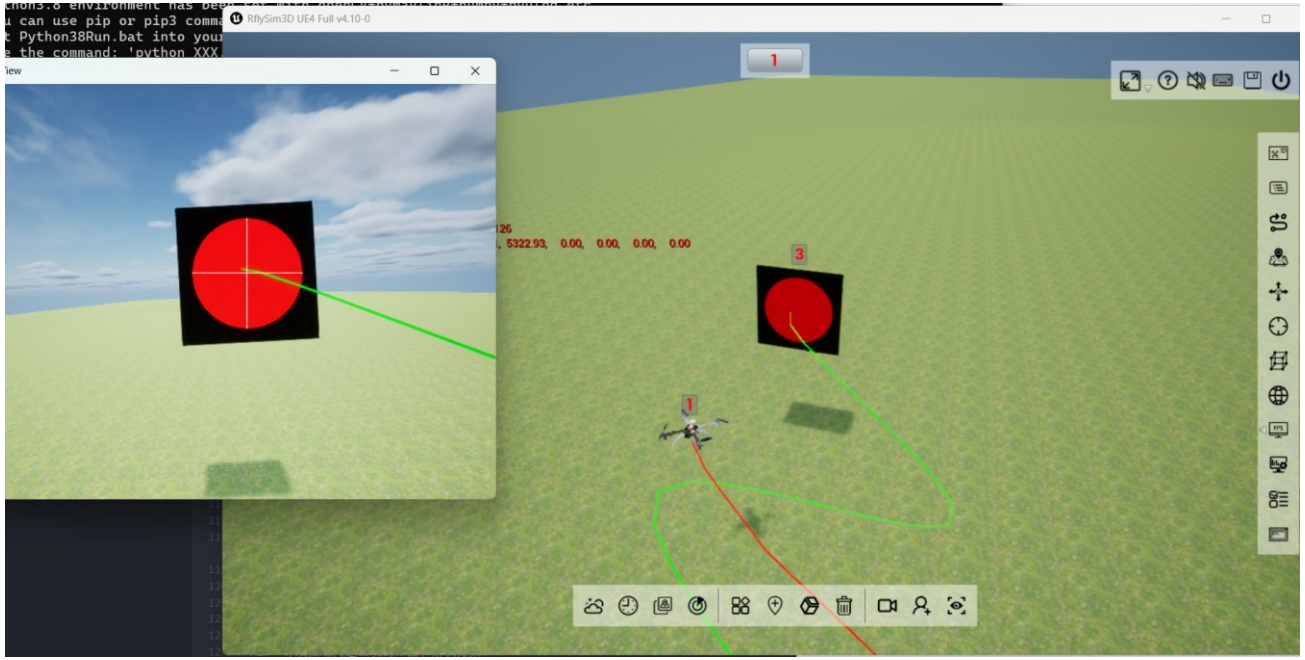
8.8.2 Basic and Integrated Experiments
Textbook: Sec. 8.3.1 and Companion Examples

8.8.3 Advanced Case Studies
Textbook: Sec. 8.3.2

8.8.4 Simulation-to-Real Transfer
Textbook: Sec. 8.2.2

8.8.2 Basic and Integrated Experiments: Plate Following

A Color Mask Yields Target Center and Size



Example Documentation: Plate and Drone View

Image/mask: 640×480; two red hue–saturation–value (HSV) bands.

Target data: Bounding-region center and box height.

Control inputs: Center errors drive lateral/vertical motion; box-height error drives approach/recession.

8.8.2 Basic and Integrated Experiments: Control Example

How Image Errors Become Velocity Commands

Given: 640×480; target center (360,260); initial/current box height 100/80 px.
P = proportional control. All three errors exceed the 15 px deadband.

$$v_x = 0.015(100 - 80) = 0.30 \ \text{m/s}$$

$$v_y = 0.009(360 - 320) = 0.36 \ \text{m/s}$$

$$v_z = 0.005(260 - 240) = 0.10 \ \text{m/s}$$

SendVelFRD(0.30,0.36,0.10,0): Heading rotates horizontal velocity; vertical stays North–East–Down (NED); yaw rate 0.
Aligned forward camera, near-level body: Smaller box → forward; right/down error → right/down.

8.8.2 Basic and Integrated Experiments: Execution Steps

Check Startup, Lock-On, and Shutdown Separately

[installation directory]/RflySimAPIs/8.RflySimVision/
1.BasicExps/1-VisionCtrlDemos/e6_Circle-follow/Readme.pdf

Start: circle_follow.bat launches software-in-the-loop (SITL; SIL in the textbook). Wait for Global Positioning System (GPS) 3D fixed and extended Kalman filter (EKF) initialization.

Control: Use the bundled Python environment for circle_follow_autoCtrl.py; use circle_follow_kbCtrl.py for one-axis tests.

Observe/finish: Check lock-on, image error, and vehicle response. Esc stops the display; then apply the vehicle shutdown policy.

BAT cleanup kills Python/CMD by name; protect other tasks on this machine.

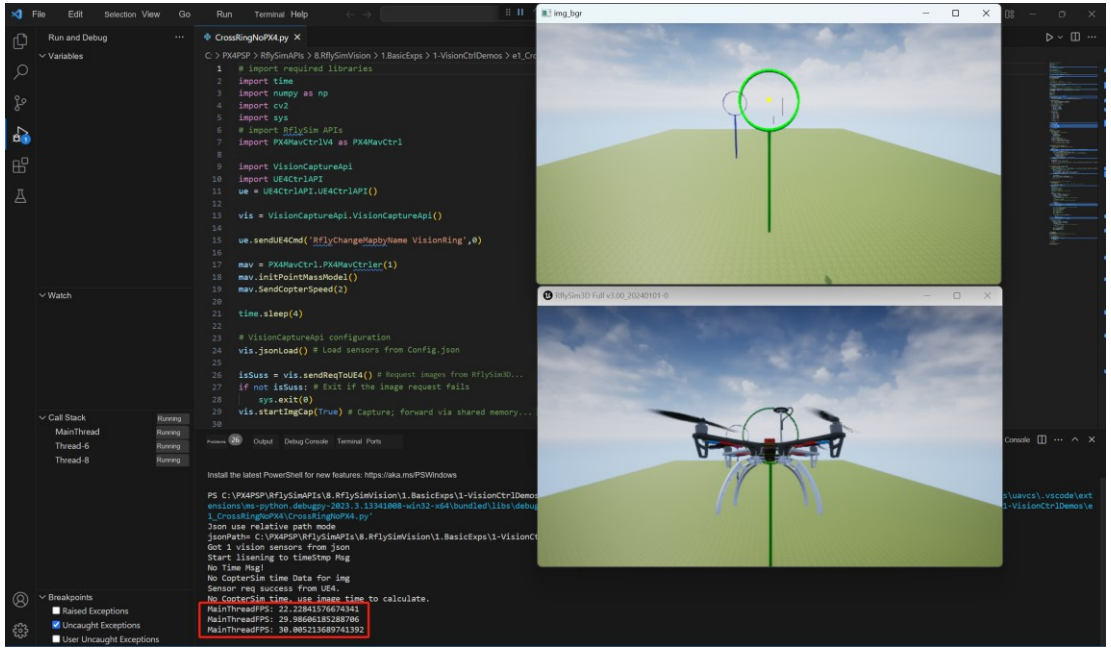
8.8.2 Basic and Integrated Experiments: Interpreting Results

Relate Image Errors to Vehicle Responses

Given Change	Source-Code Prediction	Interpretation
Center Moves Right / Down	Positive rightward/downward velocity	Assumes an aligned forward camera and near-level body
Box Height Below Initial Value	Positive forward velocity	Size is a range proxy, not metric position
Error Within 15 px Deadband	Corresponding P adjustment stops	Deadband is not measured accuracy
Target Lost / Esc Pressed	Skip command / stop display thread	Define hold, mode change, or landing separately

8.8.2 Basic and Integrated Experiments: Ring Passing

Position Selects Stages; Vision Steers Through Targets



Example Documentation: Green Ring and Onboard View

Entry: e1_CrossRingNoPX4/ CrossRingNoPX4.bat and CrossRingNoPX4.py

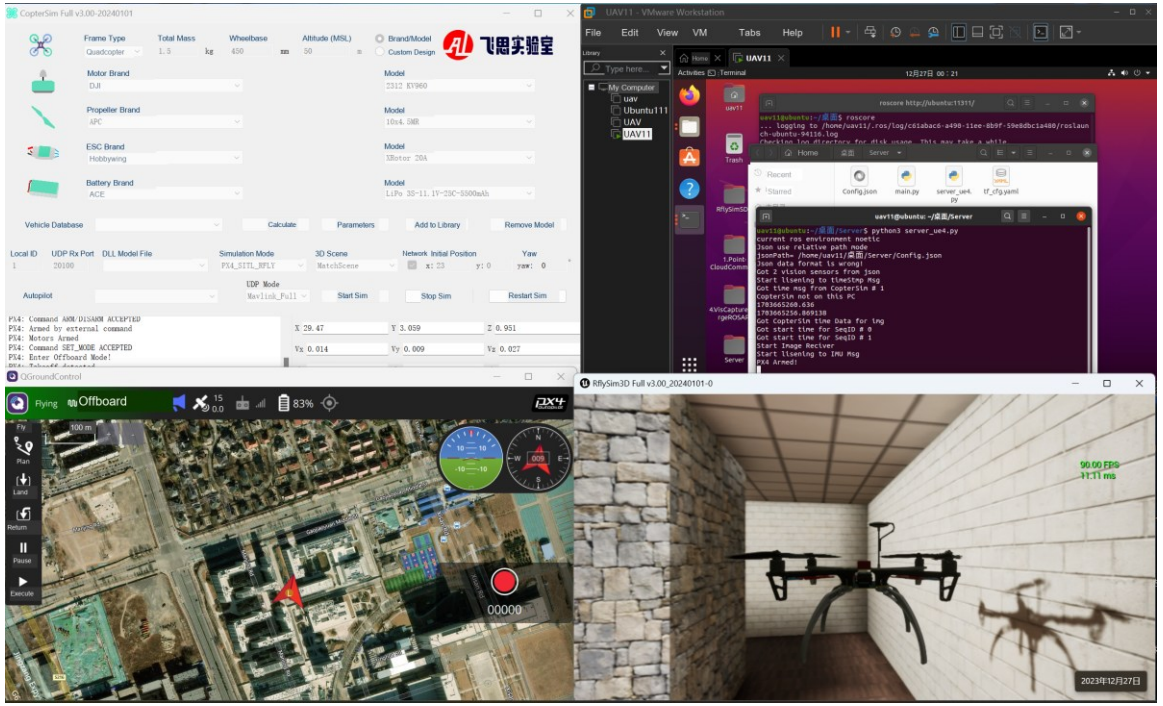
Data/task: Forward position selects the stage and detector; image errors steer vertical and yaw commands.

Check: Images, stages, and pose agree; descent and exit conditions differ.

Forward Position Selects Stage → **Stage-Specific Detector** → **Image Errors → Motion Point-Mass Pose Feedback**

8.8.2 Basic and Integrated Experiments: LiDAR SLAM

Motion Supplies Observations; SLAM Produces Pose and Maps



Example Documentation: Indoor Scene and Receiver

Target chain (integration required): LiDAR points → compatible scan → Cartographer.

Mapping/localization: Build and save a map first; then update pose within that map.

Check: Map and estimated pose remain continuous.
Waypoint flight alone does not validate SLAM.

8.8.2 Basic and Integrated Experiments: SLAM Setup and Checks

Connect Topics, TF, and Algorithm Nodes

[installation directory]/RflySimAPIs/8.RflySimVision/2.AdvExps/e6_LaserSLAMdemo/Readme.pdf

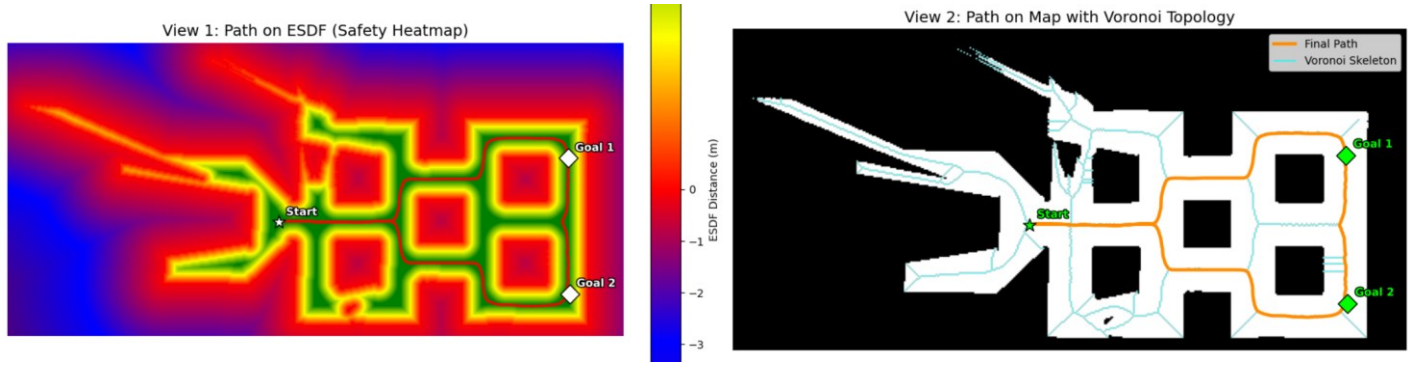
Platform/publisher: RflySimPlatform/Client/VisionCapAPIDemo.bat; run RflySimPlatform/Server/server_ue4.py on the ROS1 side.

Setup required: Select a LiDAR input. Match cloud/scan topics, LaserScan vs MultiEchoLaserScan, and TF before launching Cartographer.

Mode/exit: A task launch must load rflysim_mapping.lua or rflysim_localization.lua. Save/load .pbstream separately; then finish vehicle control.

8.8.2 Basic and Integrated Experiments: ESDF and Voronoi

An Occupancy Map Yields Obstacle Distance and Candidate Paths



Example Documentation: Distance Field, Skeleton, and Path

Occupancy map:
Distinguishes obstacles from traversable space.

Euclidean signed distance field (ESDF): Obstacle clearance.
Skeleton: Candidate corridors.

Planning output:
planned_path.npy.
Transform coordinates before execution.

8.8.2 Basic and Integrated Experiments: Path Coordinates

Row/Column Errors Send Waypoints in the Wrong Direction

Given: 0.05 m/px; start (row 100, col 200); path point (row 120, col 260).
Use the current code's row/column-to-NED mapping; set Down = −1 m.

$$N = (260 - 200) \times 0.05 = 3.0 \text{ m}$$

$$E = (120 - 100) \times 0.05 = 1.0 \text{ m}, \qquad D = -1.0 \text{ m}$$

Wrong mappings: Transposing gives North = 1, East = 3 m; reversing row direction gives East = −1 m.
Field check: Use two known landmarks to verify map origin, scale, and axis directions.

8.8.2 Basic and Integrated Experiments: Path Acceptance

Match the Current Path File to the Current Flight Trajectory

[installation directory]/RflySimAPIs/8.RflySimVision/
2.AdvExps/e16_ESDFPathPlan/Readme.pdf

Generate/transfer: In maps/WinWSL.bat, run python3 OGM2ESDF.py. Select points; copy planned_path.npy to Server.

Start/check: Client/VisionCapAPIDemo.bat; in Server/WinWSL.bat, run python3 esdf_planner.py.

Results/exit: Verify one start → goal 1 → goal 2 → start tour using measured pose and obstacle clearance. The baseline repeats; stop explicitly and save logs.
2D fixed-altitude baseline; 3D, timed trajectories, and dynamic constraints require extensions.

8.8.2 Basic and Integrated Experiments: You Only Look Once (YOLO) Following

Trained Weights Connect Data Capture to Detection and Following

[installation directory]/RflySimAPIs/8.RflySimVision/
2.AdvExps/e17_YoloDroneFollow/Readme.pdf

Capture Images and Labels → **Split Train / Val** → **Train and Save Weights** → **Detect and Follow**

Script order: get_drone_dataset.py → split_dataset.py → train_yolo.py → DroneFollow.py.

Required: Ultralytics/PyTorch and matching drone data/classes/weights. Default: Train/val only; reserve independent test scenes separately.

Check/exit: Inspect detections, selected boxes, and response. Baseline: Top-score box per frame, no persistent ID. Keep matching records; define loss/exit actions.

8.8.2 Basic and Integrated Experiments: Combined Results

Different Experiment Questions Need Different Evidence

Task Level	Required Evidence	Insufficient Substitute
Object Detection	Boxes/classes, annotation matching, statistical metrics	Raw image or platform ground-truth boxes alone
Continuous Tracking	Target identity, loss, reassociation	A target box in one frame
SLAM Localization	Map, estimated poses, continuous trajectory	Flight along preset waypoints
Planning and Control	Match planning, commands, motion from one run	Takeoff or error-free exit

Chapter Outline

Experiments and Simulation-to-Real Transfer

Chapter 8 Multimodal Perception and Intelligent Decision-Making

8.1 Image Processing Fundamentals

8.2 Recognition, Tracking, Optical Flow

8.3 Localization, Mapping, and Planning

8.4 Common Perception Sensors

8.5 Coordinate Transformations

8.6 Fusion and Control Decision-Making

8.7 Tools and Framework Interfaces

8.8 Experiments and Real-World Transfer

8.9 Exercises and Classroom Discussion

4 Units in This Section

8.8.1 Chapter Experiment List
Textbook: Sec. 8.3.1

8.8.2 Basic and Integrated Experiments
Textbook: Sec. 8.3.1 and Companion Examples

8.8.3 Advanced Case Studies
Textbook: Sec. 8.3.2

8.8.4 Simulation-to-Real Transfer
Textbook: Sec. 8.2.2

8.8.3 Advanced Case Studies: Advanced Tasks

Four Cases Extend Different System Capabilities

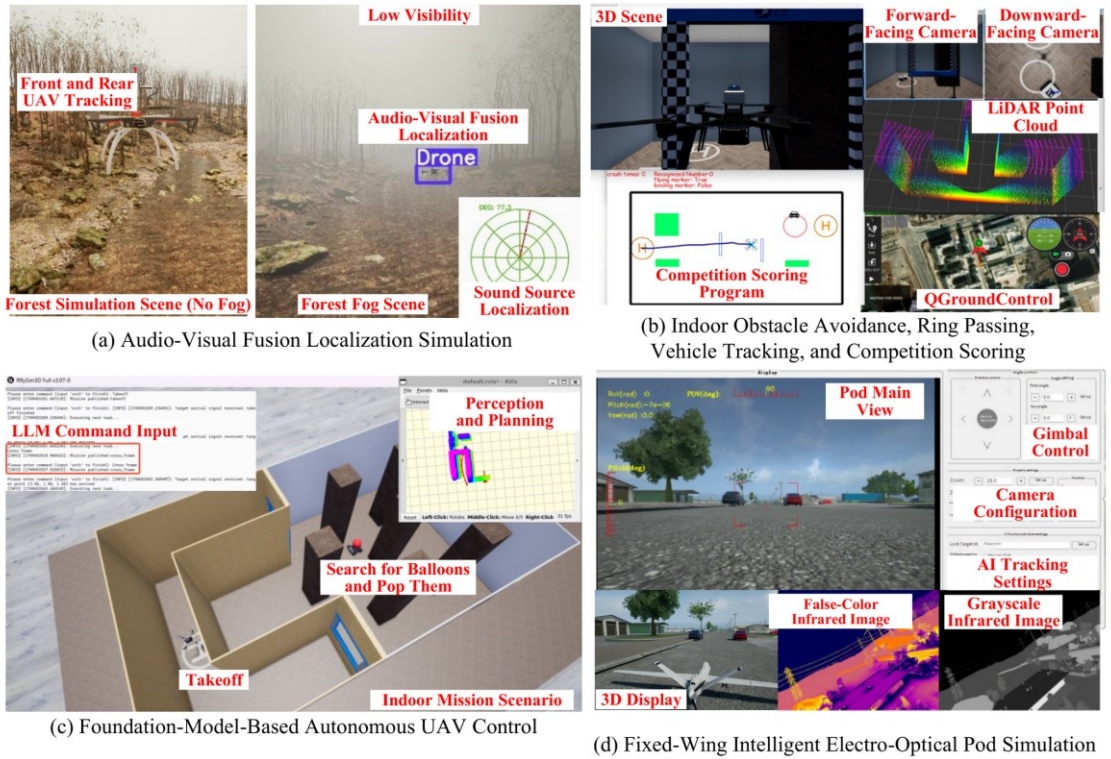


Fig. 8.12 Other Advanced Perception and Decision-Making Cases

(a) Acoustic-visual (audio-visual): Two modalities give target-bearing cues.

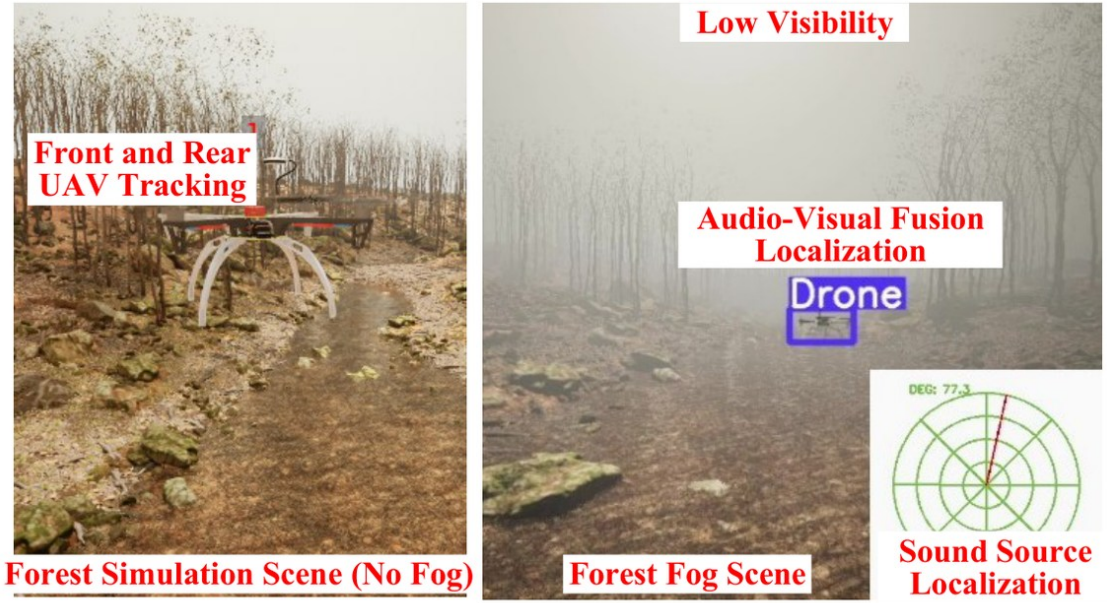
(b) Indoor tasks: Avoidance, rings, vehicle tracking, scoring.

(c) Foundation models: Visual semantics, code, task structures.

(d) Fixed-wing pod: Three-axis pointing; multiple imaging modes.

8.8.3 Advanced Case Studies: Acoustic-Visual Fusion

Sound Adds Bearing Cues When Vision Is Occluded



(a) Audio-Visual Fusion Localization Simulation

Vision: Images → boxes, classes, associated targets.

Audio: Generalized cross-correlation (GCC) gives delay → bearing. Check validity, noise, and reflections.

Example entry: 3.CustExps/ e12_ObjDetectaudio-visual/ quadrotor_follow.py

Fig. 8.12(a) Acoustic-Visual Localization Simulation

Needs Unreal Engine (UE) audio, model code/weights, and Full Edition+ with a matching license.

8.8.3 Advanced Case Studies: Acoustic Localization Example

Two-Channel Delay Constrains Source Bearing

Far-field teaching model: Mic spacing 0.20 m; sound speed 343 m/s; arrival delay 0.30 ms. Measure angle from the positive broadside direction. Find path difference, then bearing.

$$\Delta r = c\Delta t = 343 \times 0.00030 = 0.1029 \text{ m}$$

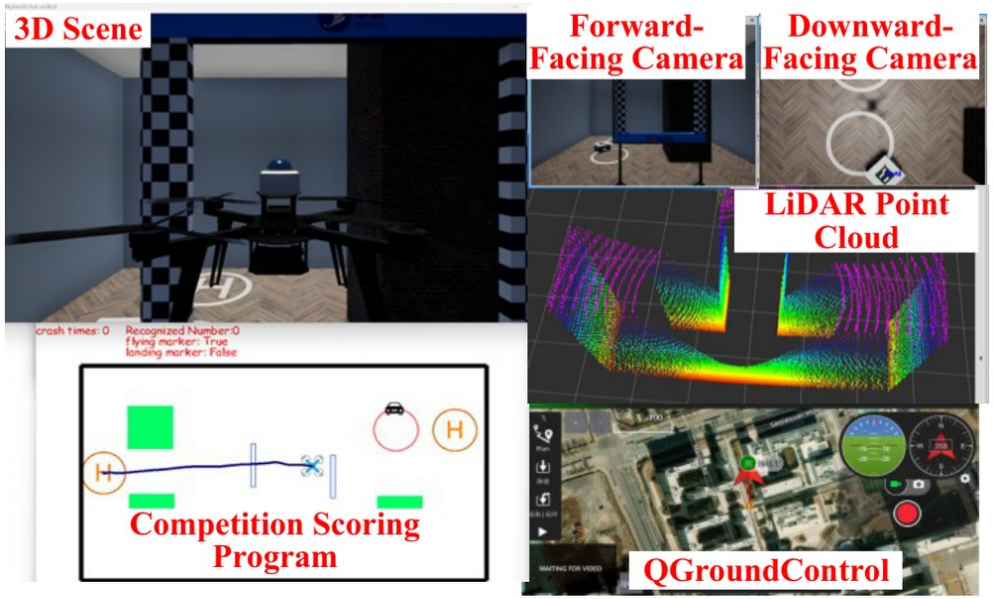
$$\theta = \arcsin\frac{\Delta r}{d} = \arcsin(0.5145) \approx 31.0^\circ$$

$$|\Delta t| \le \frac{d}{c} = \frac{0.20}{343} \approx 0.583 \text{ ms}$$

Interpretation: 0.30 ms is feasible; 31.0° constrains bearing but leaves directional ambiguity. More microphones/visual association add cues. Reflections, noise, and near-field effects need further modeling.

8.8.3 Advanced Case Studies: Indoor Integration and Scoring

Perception and Scoring Read Different Data



(b) Indoor Obstacle Avoidance, Ring Passing, Vehicle Tracking, and Competition Scoring

Fig. 8.12(b) Integrated Indoor Task and Scoring

Perception: Forward/downward cameras + LiDAR → pose, map, target data.

Execution: Avoid obstacles, pass rings, track a vehicle. A controller tracks the generated commands.

Scoring entries: 4_CompSlamNav and e11_KCFdemo. Scoring subscribes to state; keep ground truth separate from algorithm inputs.

8.8.3 Advanced Case Studies: Scoring Example

Compare the Same Task Using the Same Criteria

Teaching score: Four rings; record passes and collisions. This is not an official competition rule.
Assess constraints separately: Collision-free operation, tracking error, and completion time.

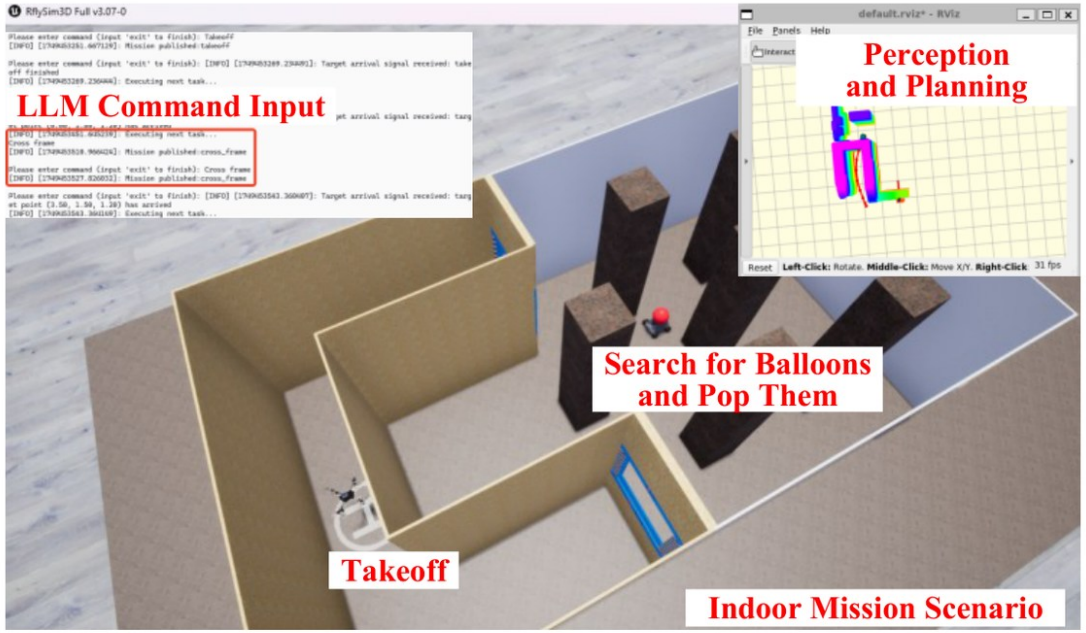
$$S = 100\frac{n_{\text{pass}}}{4} - 20n_{\text{collision}}$$

Given Design	Teaching Score	Interpretation
A: 4/4 Passed, 1 Collision	100−20=80	High completion; violates the no-collision condition
B: 3/4 Passed, 0 Collisions	75−0=75	No collision; one ring remains incomplete

Conclusions use declared metrics; report task time and trajectory or target-tracking error separately.

8.8.3 Advanced Case Studies: Foundation-Model Tasks

Language Forms Task Structures Before Calling Skills



(c) Foundation-Model-Based Autonomous UAV Control

Fig. 8.12(c) Foundation-Model Autonomous Task Example

Three roles: Visual semantics; code generation; task-command or behavior-tree generation.

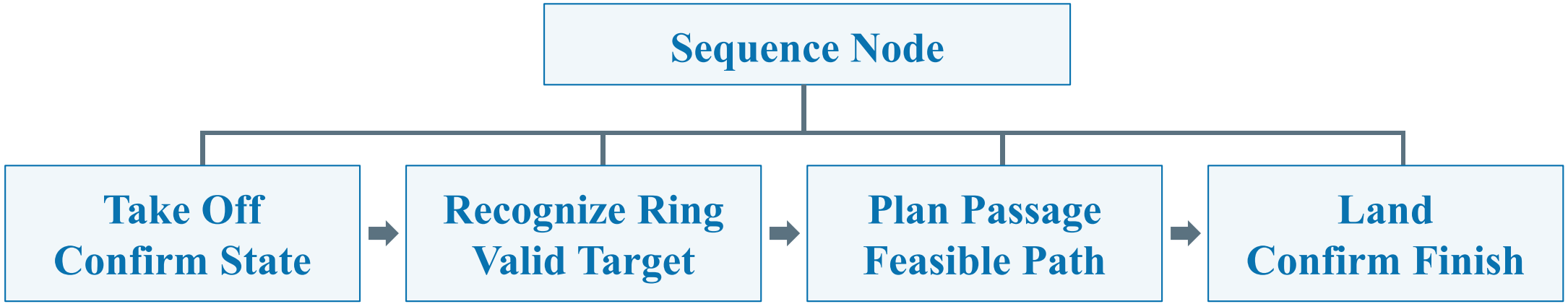
Execution: Validate structure → call skills → read feedback. Keep fast control independent.

1.BasicExps/
5.LLMUavComp/
6.LLMUavCompSLAM/
Readme.pdf in Each Folder

Both entries: 1.BasicExps/. Match runtimes, model services, and containers to Readme.pdf.

8.8.3 Advanced Case Studies: Behavior Execution

Structured Tasks Preserve Conditions, Actions, and Failure Branches



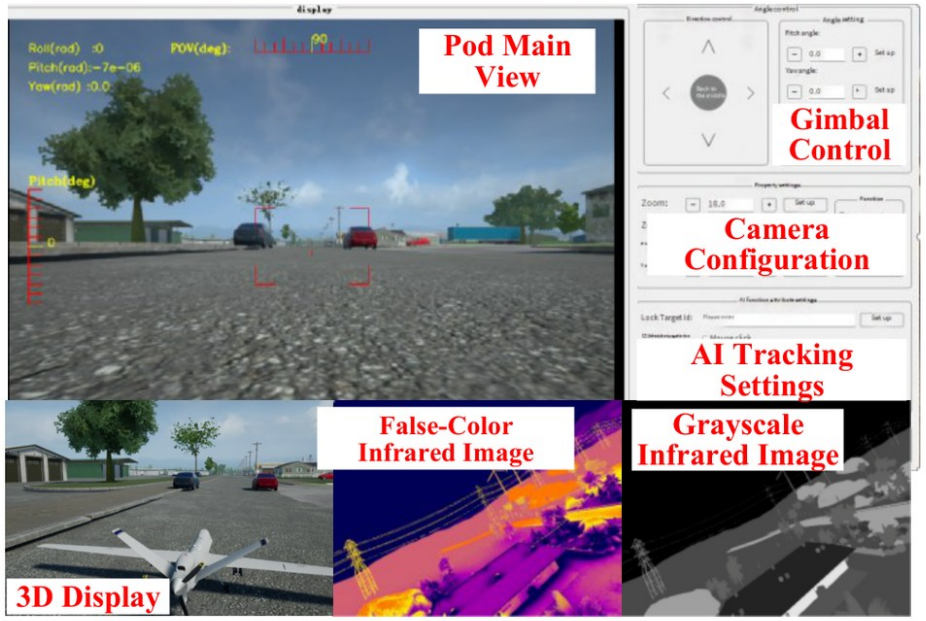
Left to right: Advance on SUCCESS; wait on RUNNING; return immediately on FAILURE.
Check fields, skill names, parameters, bounds first; read real feedback during execution.

Outer Recovery: Reobserve or Terminate

Language generates tasks; feedback determines whether execution continues.

8.8.3 Advanced Case Studies: Fixed-Wing Gimbal Pods

Gimbal and Vehicle Attitude Change the Line of Sight Separately



(d) Fixed-Wing Intelligent Electro-Optical Pod Simulation

Fig. 8.12(d) Fixed-Wing Intelligent Electro-Optical Pod

Three observations: Visible light, infrared, ranging. Distinguish imaging type from range.

Two angle sets: Vehicle attitude and three-axis gimbal angles. Zoom changes field of view.

0.ApiExps/12.UAV_PodSimUI/ AircraftMathworksSITLRun.bat; Python38Run.bat → main.py.

8.8.3 Advanced Case Studies: Gimbal Readout Example

How Much Line-of-Sight Error Does a Pixel Offset Represent?

Pinhole teaching model: Image width 640 px; target horizontal offset 40 px.
Horizontal field of view changes from 90° to 60°; the same pixel offset gives a different angle.

$$f_x = \frac{W}{2\tan(\mathrm{FOV}_x/2)}, \qquad \delta = \arctan\frac{40}{f_x}$$

Horizontal Field of View	Focal Length (px)	Angle for a 40 px Offset
90°	320.0	7.13°
60°	554.3	4.13°

After zooming, update imaging parameters and control angular error. Box size is not metric length.

Chapter Outline

Experiments and Simulation-to-Real Transfer

Chapter 8 Multimodal Perception and Intelligent Decision-Making

8.1 Image Processing Fundamentals

8.2 Recognition, Tracking, Optical Flow

8.3 Localization, Mapping, and Planning

8.4 Common Perception Sensors

8.5 Coordinate Transformations

8.6 Fusion and Control Decision-Making

8.7 Tools and Framework Interfaces

8.8 Experiments and Real-World Transfer

8.9 Exercises and Classroom Discussion

4 Units in This Section

8.8.1 Chapter Experiment List
Textbook: Sec. 8.3.1

8.8.2 Basic and Integrated Experiments
Textbook: Sec. 8.3.1 and Companion Examples

8.8.3 Advanced Case Studies
Textbook: Sec. 8.3.2

8.8.4 Simulation-to-Real Transfer
Textbook: Sec. 8.2.2

8.8.4 Simulation-to-Real Transfer: Framework

Five Stages Connect Data, Models, and System Validation

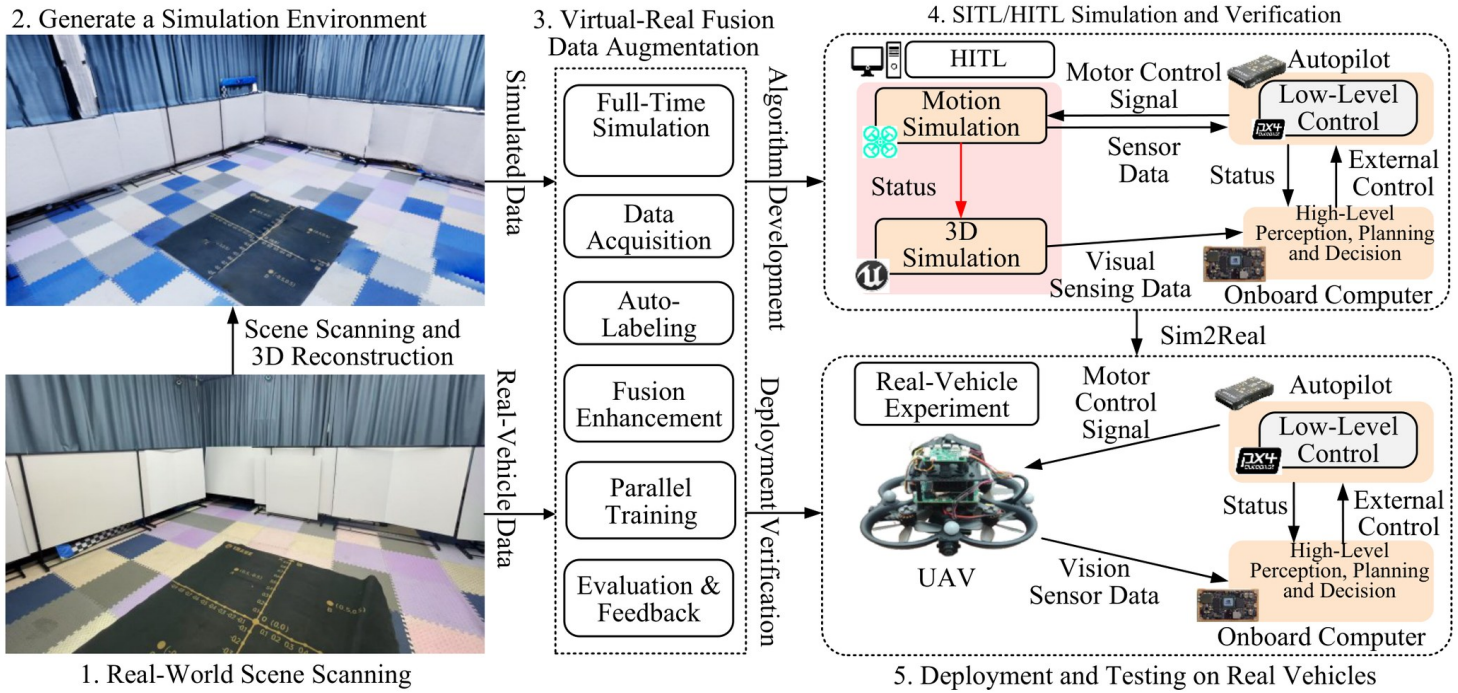


Fig. 8.10 Simulation-to-Reality Development Framework

Scan → model → augment → SITL/HITL → real deployment; feed validation back.

SITL/HITL: Software-/hardware-in-the-loop; textbook abbreviations SIL/HIL.

8.8.4 Simulation-to-Real Transfer: Fig. 8.10 Left Detail

Scanning and Augmentation Connect Real Scenes to Training

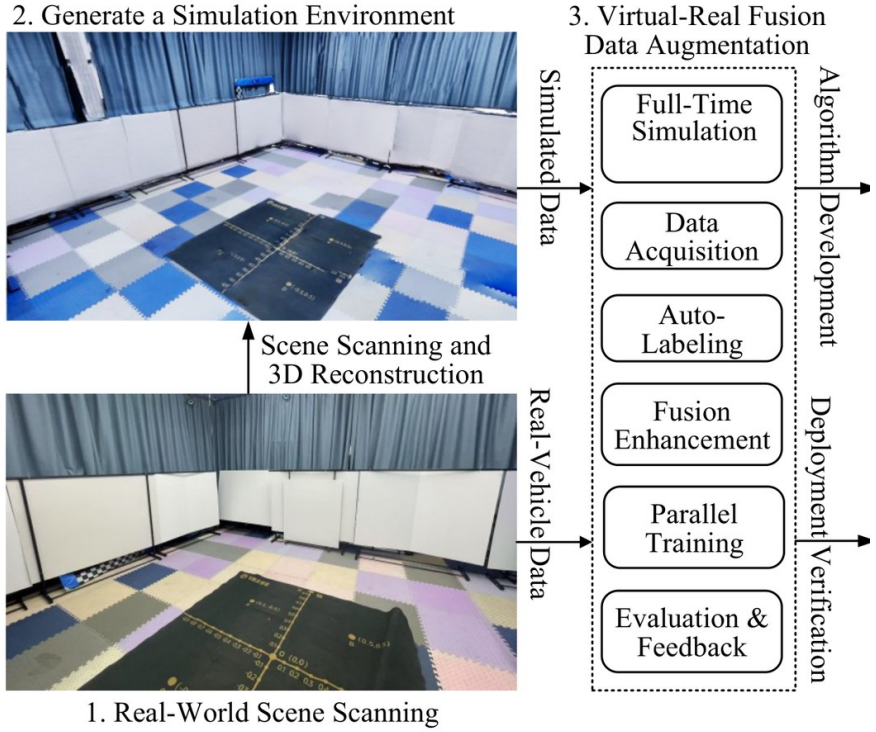


Fig. 8.10 Detail: Scanning, Scenes, and Augmentation

1. Real scanning: Capture geometry/appearance; retain scale and reference frame.

2. Simulation: Rebuild obstacles, meshes, textures; generate controlled conditions.

3. Simulated/real augmentation: Capture, label, fuse, train; use evaluation feedback.

Two data streams share one training format. An independent real test set measures transfer.

8.8.4 Simulation-to-Real Transfer: Fig. 8.10 Right Detail

The Same High-Level Algorithm Moves From In-Loop to Real Execution

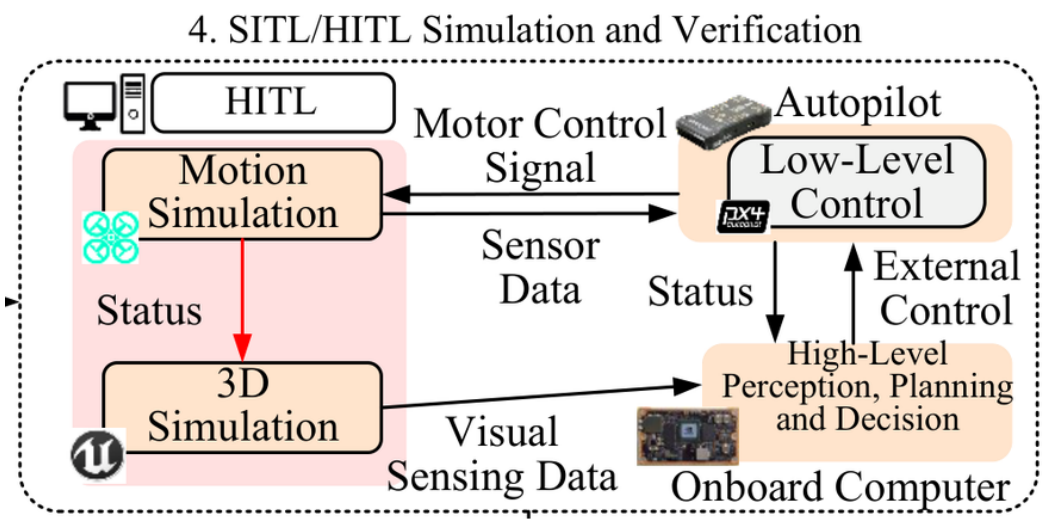


Fig. 8.10 Stage 4: In-Loop Tests

In-loop: Real computer + flight controller.

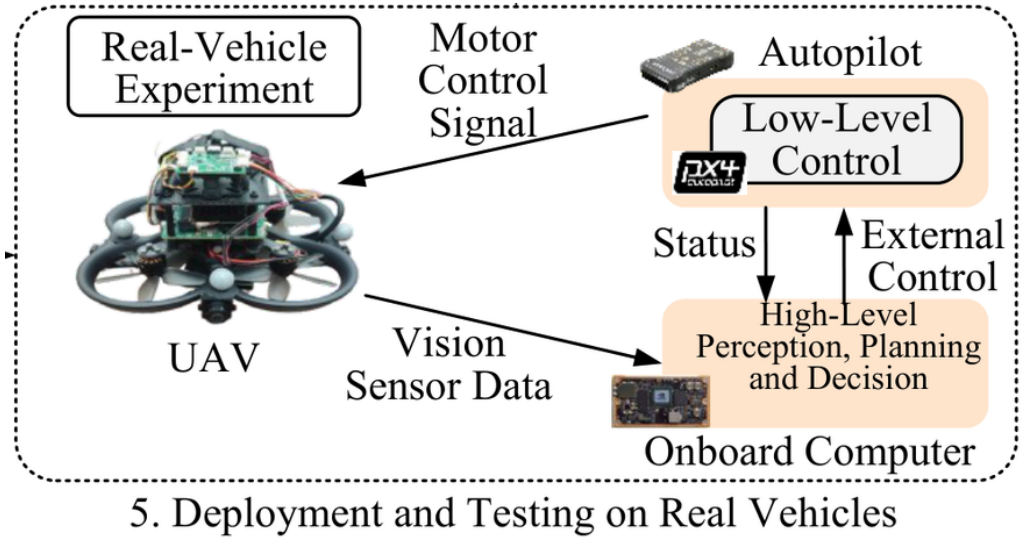
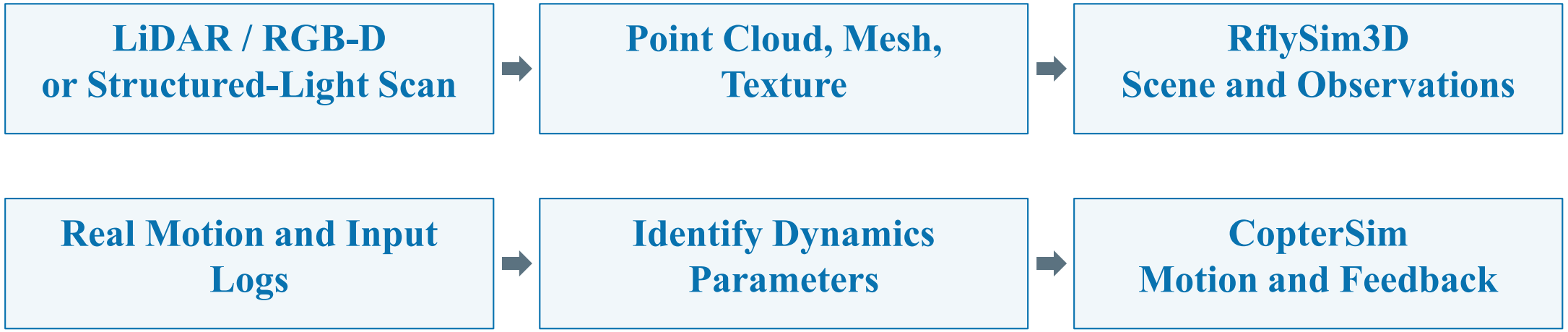


Fig. 8.10 Stage 5: Real Deployment and Experiments

Real vehicle, observations, motion response.

8.8.4 Simulation-to-Real Transfer: Scanning and Reconstruction

Geometric Twins and Dynamics Models Solve Different Problems



Geometry check: Obstacle positions, scale, collision boundaries, and viewing geometry agree.

Dynamics check: Comparable inputs give comparable attitude, velocity, and position responses. Appearance cannot replace motion checks.

8.8.4 Simulation-to-Real Transfer: Data Augmentation

Varied Samples Expand Coverage; Real Data Measure Transfer

Varied Simulation Automatic Labels → **Unified Data/Label Format** → **Training and Validation Fixed Real Test Set**

Expand coverage: Day/night, weather, target pose, occlusion, materials; match task classes and labels.

Fuse data: Add hard real cases and noise; unify image size, class list, labels, preprocessing.

Evaluate transfer: Split by scene; keep real test data independent. Report hits, misses, false detections.

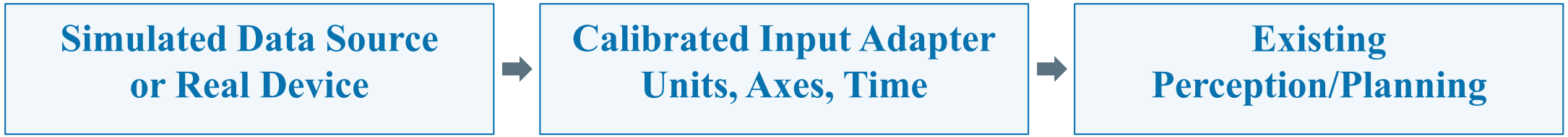
8.8.4 Simulation-to-Real Transfer: In-the-Loop Tests

Isolate Algorithm and Hardware Bottlenecks by Stage

Stage	Retained / Replaced	Problems It Can Reveal
SITL	Algorithm and autopilot software Simulated motion and observations	Logic, configuration, interfaces, task outputs
HITL	Real onboard computer and flight controller Connected to simulation	ARM target builds, device libraries, latency and communication
Real Deployment	Real sensors and vehicle Connected to the tested algorithm	Actual noise, calibration, actions, task results

8.8.4 Simulation-to-Real Transfer: Sensors and Calibration

Identical Message Structures Need Identical Physical Meaning



Camera: Intrinsics, distortion, mounting extrinsics; resolution and BGR/RGB channel order must match the model.

Depth/points: Convert valid raw values to meters; keep invalid values unknown. Define axes, origin, scan timing.

Inertial/control: Check inertial measurement unit (IMU) units, axes, camera timing. The output adapter converts target coordinates and flight-controller interfaces.

8.8.4 Simulation-to-Real Transfer: Compute Budget

Fast Individual Stages Can Still Miss a Serial Deadline

Teaching conditions: Camera 20 Hz; the serial chain must finish within 50 ms per frame.

Stage	Given Time	Affected Output
Capture + Preprocess	8 + 5 ms	Frame transfer and tensor preparation
Inference + Planning	26 + 12 ms	Target data and trajectory computation
Send Control	6 ms	Action enters the execution interface

$$T = 8 + 5 + 26 + 12 + 6 = 57 \text{ ms} > 50 \text{ ms}$$

8.8.4 Simulation-to-Real Transfer: Time, Communication, and Control

Align Time and Coordinates Before Comparing Actions



Time/network: Confirm stamp is sample time; use one time base. Measure delay, packet loss, queuing.

Map/position: Transform local SLAM and global navigation satellite system (GNSS) constraints. Confirm East–North–Up (ENU) and NED axes, origins, units.

Execution/exit: Send setpoints through the correct interface and maintain external-control timing. On failure/exit, switch mode or execute the landing policy.

8.8.4 Simulation-to-Real Transfer: Paired Comparisons

Compare Hardware, Detection, and Trajectories

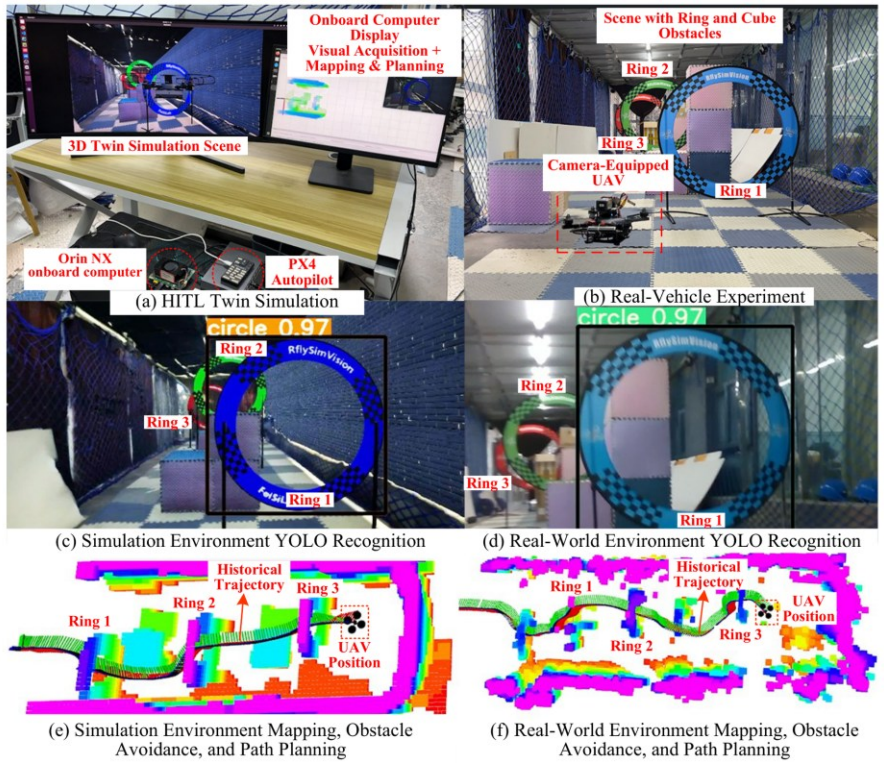


Fig. 8.11 Paired Simulation-to-Reality Comparison

(a,b) System: Onboard computer, PX4, vehicle.

(c,d) Detection: Simulated and real images of the same target.

(e,f) Map/trajectory: Estimates, plans, responses.

8.8.4 Simulation-to-Real Transfer: System Comparison

In-Loop Hardware Carries Over to the Real UAV

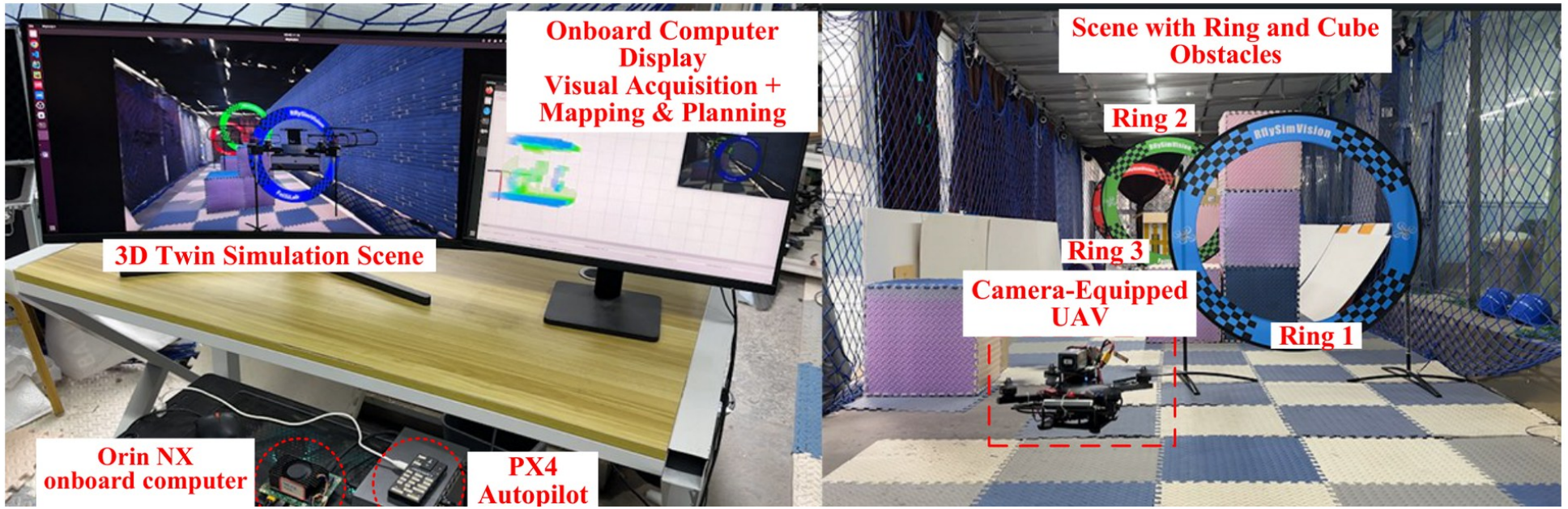


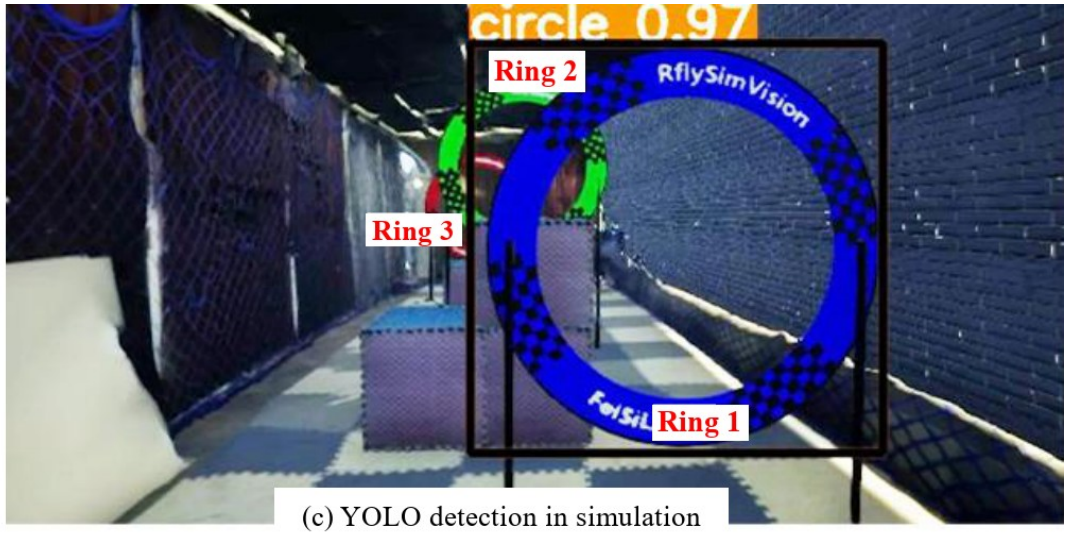
Fig. 8.11(a,b) In-Loop System and Real UAV

Orin NX: High-level tasks; PX4: Low-level control on the unmanned aerial vehicle (UAV). Match its architecture.

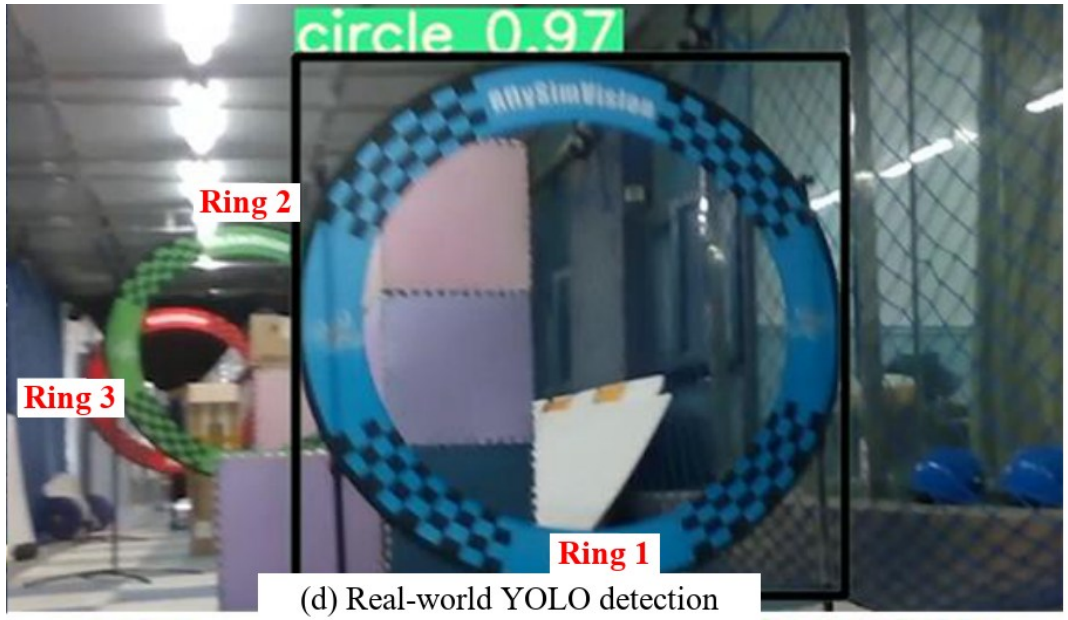
Real vehicles/sensors supply motion/observations. Recheck calibration, communication, and deadlines.

8.8.4 Simulation-to-Real Transfer: Detection Comparison

Two Detection Images Cannot Replace Dataset Metrics



(c) YOLO detection in simulation



(d) Real-world YOLO detection

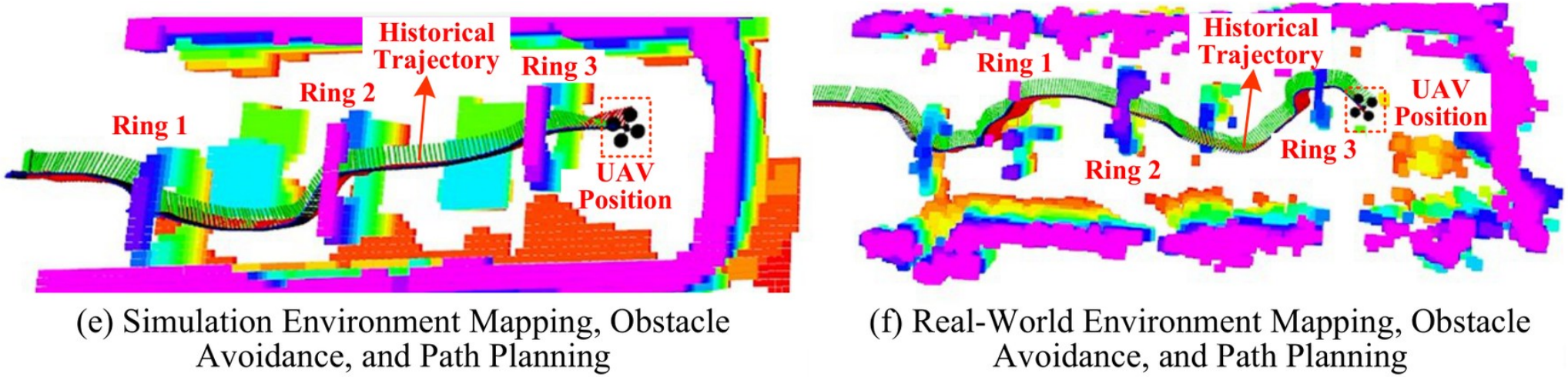
Fig. 8.11(c,d) Simulated and Real Ring Detection

Per-image: Match box/class to labels. A value such as 0.97 is confidence.

Dataset: Count hits, false detections, misses; group by real lighting, background, and blur.

8.8.4 Simulation-to-Real Transfer: Mapping and Planning

Similar-Looking Paths Still Need Quantitative Task Results



(e) Simulation Environment Mapping, Obstacle Avoidance, and Path Planning

(f) Real-World Environment Mapping, Obstacle Avoidance, and Path Planning

Fig. 8.11(e,f) Maps, Plans, and Trajectories

Read separately: Spatial map, estimated pose, planning reference, historical trajectory.

Same start/goals/obstacles: Quantify success, collisions, tracking error, completion time, and latency.

8.8.4 Simulation-to-Real Transfer: Validation Example

Report Transfer Differences Under Matched Test Conditions

Teaching inputs: 20 trials per setting; same start, goals, obstacles. Success = completion with constraints met.
Simulation: 19 successes; real: 17. Keep collision, latency, and error records.

$$P_{\text{sim}} = \frac{19}{20} = 95\%, \qquad P_{\text{real}} = \frac{17}{20} = 85\%$$

Statistic	Given Result	What It Shows
Success-Rate Gap	10 percentage points	Transfer difference under fixed task conditions
Failure / Delay / Error	Group each trial	Locate data, planning, or execution problems

Check interfaces, compare full tasks, then expand operating conditions.

Exercises and Discussion

Propose Experiment and Transfer Evidence for Indoor Ring Passing

1. Entries/evidence: For indoor avoidance and ring passing with forward/downward cameras and LiDAR, which interfaces and task entries would you choose? Which algorithm and response outputs would you save?

2. Deadline: Camera 20 Hz; capture 8, preprocess 5, inference 26, planning 12, send 6 ms. Does the chain meet the period? Suggest an improvement.

3. Evaluation/exit: Both simulation and real windows show 0.97. Does that mean 97% accuracy? What evidence is needed? What must vehicle control do at exit?

Suggested Answers

Explain Results From Interfaces Through Task Execution

Entries/outputs: 0.ApiExps capture, ROS, TF; 4_CompSlamNav integrated task.
Save observations, poses, maps, paths, commands, responses.

Deadline: 57 ms exceeds 50 ms. Optimize large stages, reduce copies or bound queues, then remeasure end-to-end delay.

Evaluation/exit: 0.97 is confidence. Evaluate detection on a fixed real test set. Define mode, zero-velocity, or landing actions, then clean resources and save records.

Section Summary

Experiment Outputs and Transfer Claims Need Different Evidence

Experiment system: Interfaces, basic tasks, advanced integration, custom extensions. Inputs, outputs, and task products guide selection.

Advanced capabilities: Acoustic-visual bearing; indoor scoring; language-organized skills; pod line of sight separate from vehicle attitude.

Five-stage transfer: Scan → model → augment → SITL/HITL → real deployment. Compare matched tasks and varied-condition statistics.

Common errors: Ground truth as prediction; threshold as accuracy; confidence as accuracy; closing windows as vehicle shutdown. Continue in 8.9.