Intelligent Unmanned System Control Practice

Chapter 6 External Control and Trajectory Planning

Author: Xunhua Dai, Central South University

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

Email: dai.xh@csu.edu.cn

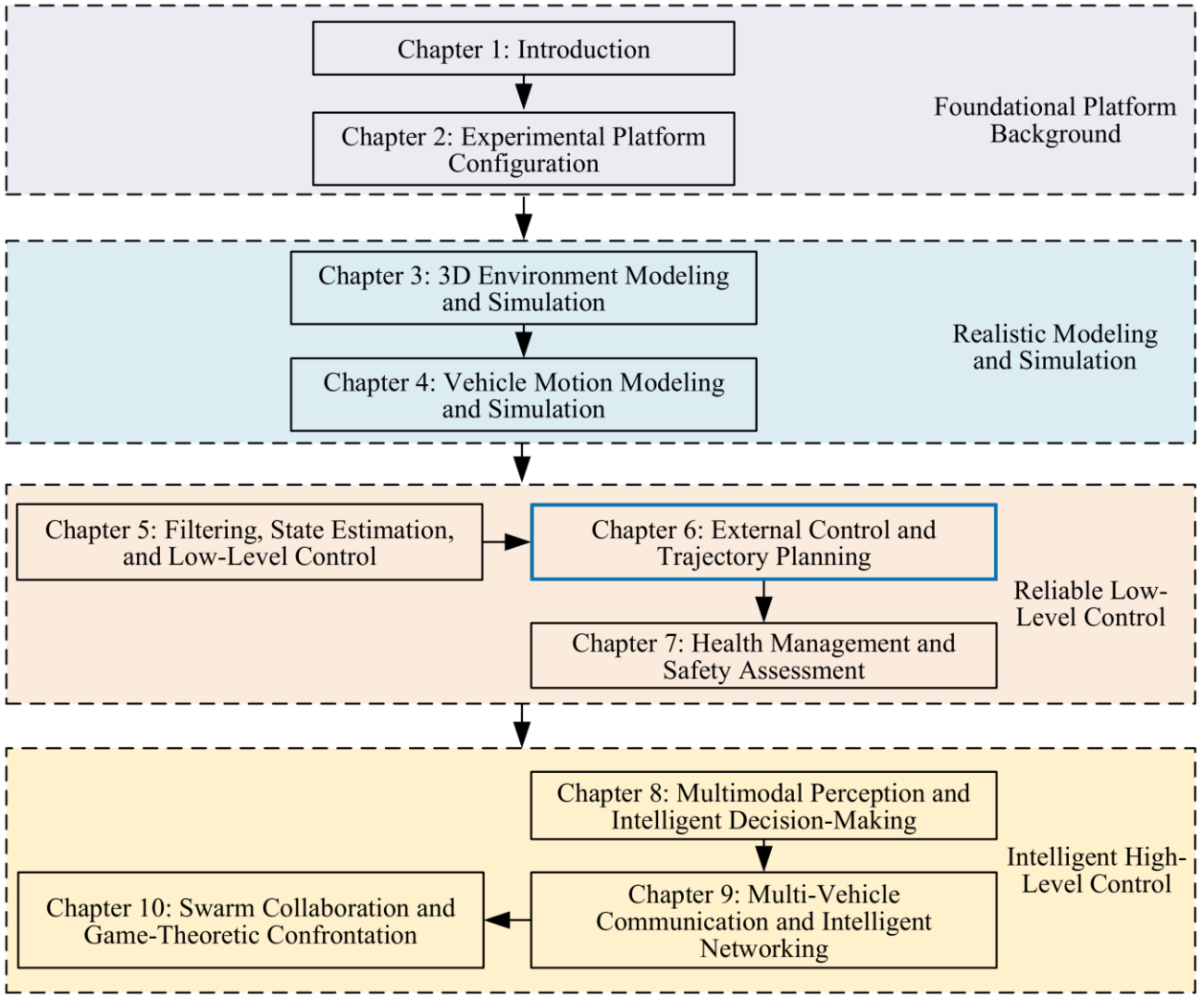
RflySim: http://rflysim.com

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 6

1. Explain control layers, command sources, and the roles of three control modes
2. Connect external control interfaces, send motion targets, and receive state feedback
3. Generate trajectories that meet timing, motion-limit, and obstacle-avoidance requirements
4. Compare reference and actual motion, explain errors, and verify the final task state

Chapter Outline

External Control and Trajectory Planning: Chapter Roadmap

Chapter 6
External Control and Trajectory Planning

6.1 External Control Fundamentals
6.2 Control Principles and Modes
6.3 Trajectory Models and Generation
6.4 Planning and Obstacle Avoidance
6.5 Tracking and Aggressive Flight Control
6.6 Frontier Control Technologies
6.7 Platform and Communication Interfaces
6.8 Examples and Experimental Validation
6.9 Exercises and Classroom Discussion

4 Units in This Section

6.1.1 Background and Significance
Textbook: Sec. 6.1.1.1

6.1.2 Control Layering
Textbook: Sec. 6.1.1.2

6.1.3 Command Sources and Motion References
Textbook: Secs. 6.1.1.3 and 6.1.3

6.1.4 Performance Analysis
Textbook: Sec. 6.1.1.4

6.1.1 Background and Significance: Tasks and Motion

Why Does Stable Hovering Not Yet Enable Autonomous Inspection?

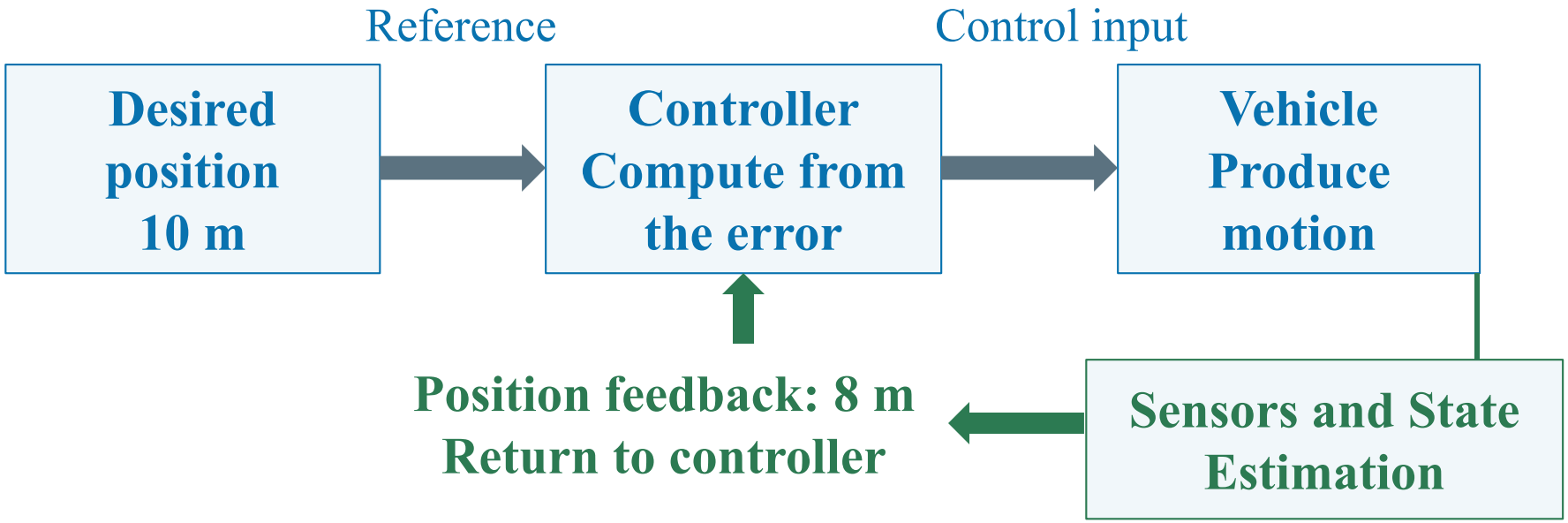
Inspection task: approach the facility, capture images along it, reroute around obstacles, then return and land.

Stable Response	Execute Motion	Complete the Task
Hold altitude and attitude Reject wind disturbances and deviations	Choose positions to pass through Schedule speed and arrival times	Check whether imaging is complete Handle changes and select the next action

Stable control is the starting point; autonomous tasks also require goals, environment, and motion to be organized.

6.1.1 Background and Significance: Closed-Loop Control

What Are the Reference, Feedback, and Control Input?



Reference: desired quantity.
Feedback: current state estimate.
Control input: quantity driving motion.

All quantities share the ground frame; new observations continually update the error.

$$e = x_{\text{ref}} - \hat{x} = 10 - 8 = 2\,\text{m}$$

e is the error, x_{ref} the desired position, and $\hat{x}$ the estimated position.

6.1.1 Background and Significance: Value of External Control

Retain Stable Inner Loops and Extend Task Capabilities

External control: modules outside the existing low-level control system (e.g., an autopilot) compute control inputs. Ground stations, onboard computers, or simulation platforms send them over wired or wireless links for low-level execution.

Task execution: turn facility inspection into a target sequence, considering efficiency, safety constraints, and state feedback.

More computing: use onboard or ground resources for image processing, planning, and learning algorithms.

Reuse existing estimation and control loops and task logic; adapt interfaces and motion limits.

Development: inspect planning, commands, and responses separately to simplify simulation comparisons and algorithm replacement.

Interfaces link task algorithms to vehicles and return state feedback and execution constraints.

Chapter Outline

External Control and Trajectory Planning: Chapter Roadmap

Chapter 6
External Control and Trajectory Planning

6.1 External Control Fundamentals
6.2 Control Principles and Modes
6.3 Trajectory Models and Generation
6.4 Planning and Obstacle Avoidance
6.5 Tracking and Aggressive Flight Control
6.6 Frontier Control Technologies
6.7 Platform and Communication Interfaces
6.8 Examples and Experimental Validation
6.9 Exercises and Classroom Discussion

4 Units in This Section

6.1.1 Background and Significance
Textbook: Sec. 6.1.1.1

6.1.2 Control Layering
Textbook: Sec. 6.1.1.2

6.1.3 Command Sources and Motion References
Textbook: Secs. 6.1.1.3 and 6.1.3

6.1.4 Performance Analysis
Textbook: Sec. 6.1.1.4

6.1.2 Control Layering: Layered Architecture

Low-Level, High-Level, and Swarm Layers Address Different Scales

Commands / motion references ↓

State feedback ↑

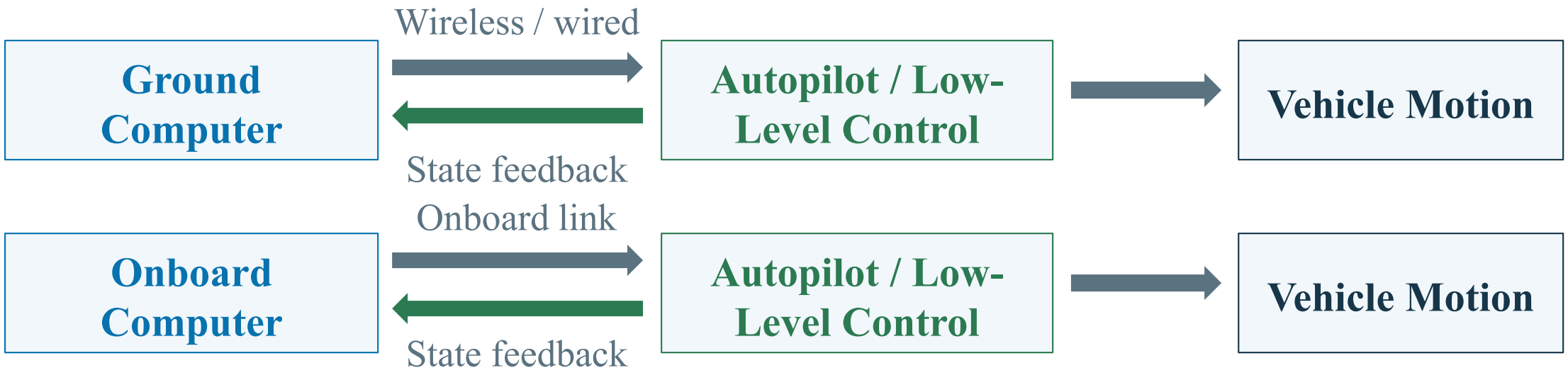
Swarm Layer	Global task, vehicle states → Task allocation and coordination constraints	**Inspection:** assign facilities to vehicles; coordinate timing, safe spacing, and communication.
High-Level Layer	Task, environment, own state → Route and motion references	**Inspection:** select approach and detour routes and decide where to capture images.
Low-Level Layer	Motion references, state estimates → Actuator control inputs	**Inspection:** track motion requirements; continually correct attitude and velocity errors.

Functions can be layered, but feedback must remain connected: every layer needs to know whether its requests were executed.

6.1.2 Control Layering: Functions and Deployment

"External" Refers to the Control Boundary, Regardless of Distance

Boundary: the existing low-level control system (defined in 6.1.1), regardless of the computer's distance from the vehicle.



Onboard does not imply low-level, nor ground-based imply swarm-level; identify responsibilities before deployment.

6.1.2 Control Layering: Typical Combinations

Combine Control Layers to Meet Task Needs

Combination	Suitable Task Organization	Still Check
Low-level only	Respond to remote or preset targets, e.g., aerial photography	Can the operator or task logic issue commands in time?
Low-level + high-level	Single-vehicle perception and planning, e.g., intelligent vehicles	Is perception valid and local computation timely?
Low-level + swarm	Central scheduling, e.g., light-show formations	Are the center, communication, and vehicle feedback available?
All three layers	Global coordination and local autonomy, e.g., rescue	Do local decisions respect global constraints?

This three-layer example is a distributed swarm; layers alone do not determine centralized or distributed deployment.

6.1.2 Control Layering: Task Levels and Interface Granularity

Closer to Actuators, External Programs Take On More Control

Command Level	Example Input	What Must the Receiver Still Do?
Swarm level	Assign three vehicles to inspect three areas	Allocate tasks, coordinate timing, and resolve conflicts
Task level	Approach the facility, image along it, then return	Assess the environment, select routes, and generate motion references
Execution level	Position, velocity, attitude, or actuator control inputs	The chosen entry point determines which internal control loops remain

For a multicopter, a position interface retains internal position and other loops; an attitude interface requires more external motion control.

Chapter Outline

External Control and Trajectory Planning: Chapter Roadmap

Chapter 6
External Control and Trajectory Planning

6.1 External Control Fundamentals

6.2 Control Principles and Modes

6.3 Trajectory Models and Generation

6.4 Planning and Obstacle Avoidance

6.5 Tracking and Aggressive Flight Control

6.6 Frontier Control Technologies

6.7 Platform and Communication Interfaces

6.8 Examples and Experimental Validation

6.9 Exercises and Classroom Discussion

4 Units in This Section

6.1.1 Background and Significance
Textbook: Sec. 6.1.1.1

6.1.2 Control Layering
Textbook: Sec. 6.1.1.2

6.1.3 Command Sources and Motion References
Textbook: Secs. 6.1.1.3 and 6.1.3

6.1.4 Performance Analysis
Textbook: Sec. 6.1.1.4

6.1.3 Command Sources and Motion References: External Control Framework

Human Task Intent Reaches the Vehicle Through Different Entry Points

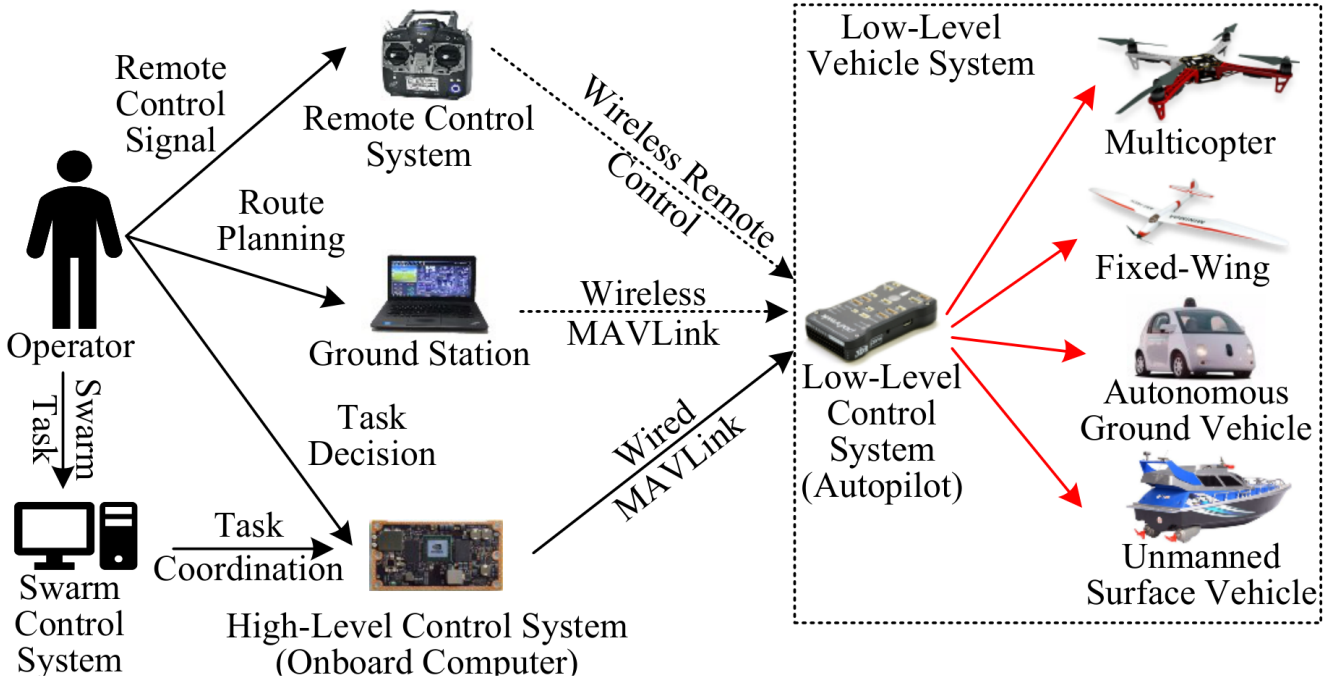


Figure 6.1 External Control Framework for Unmanned Systems

MAVLink: a common message protocol for unmanned vehicles.

Real-time control (remote signals): the operator acts continuously and judges from feedback.

Mission mode: the operator sets waypoints for task logic to execute ("Route Planning" in the figure).

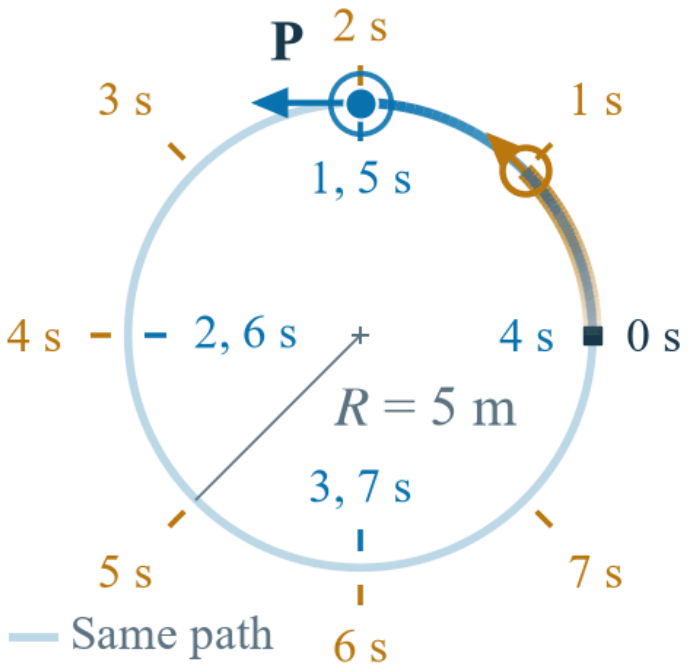
Local autonomy (task decisions): the operator sets goals and constraints; the program updates motion requirements from observations.

Vehicles at right share the same layered approach.

6.1.3 Command Sources and Motion References: Paths and Trajectories

One Path Can Have Different Trajectories

One path, two time schedules



$t = 1.00$ s

	Speed v	Accel. v^2/R
● A	7.85 m/s	12.34 m/s^2
○ B	3.93 m/s	3.08 m/s^2

First at P

A 1.00 s B 2.00 s

Marks: arrival times

Path: which positions to pass through,
with no required arrival times.

Trajectory: specifies when to reach each position.

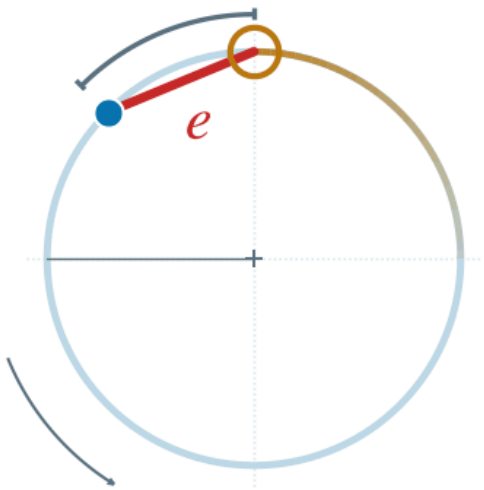
Timing: one route can require different velocities and accelerations. R is radius; t is elapsed time.

The same route can impose different motion requirements; timing is also part of planning.

6.1.3 Command Sources and Motion References: Following and Tracking

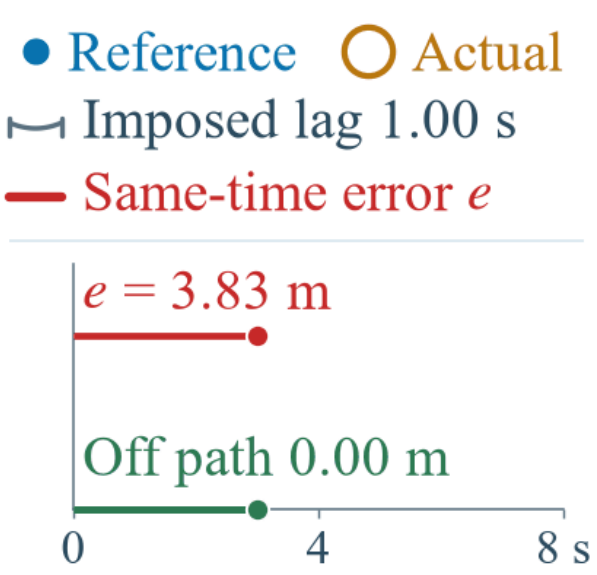
Being on the Path Does Not Mean Arriving on Time

On the path, late in time



$R = 5$ m, $v = 3.93$ m/s

$t = 3.00$ s



Path following: keep the actual position close to the desired route.

Trajectory tracking: compare reference and actual states at the same time.

Circle: both points stay on the circle. v is tangential speed; e is their simultaneous straight-line distance.

A spatial route plot can hide timing errors; tracking evaluation must align times.

6.1.3 Command Sources and Motion References: Path Feasibility

If the Centerline Clears an Obstacle, Can the Vehicle Safely Pass?

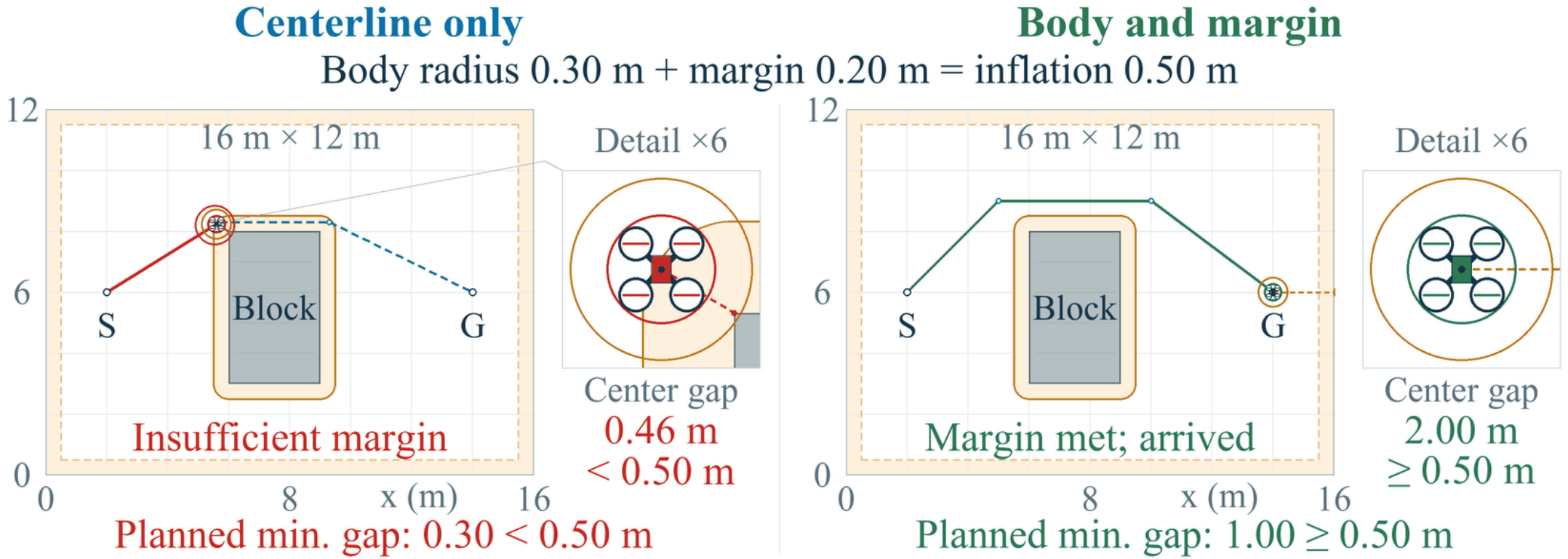


Figure: include the vehicle envelope and safety margin in route clearance

6.1.3 Command Sources and Motion References: Path Search

How Can We Find a Detour on the Same Map?

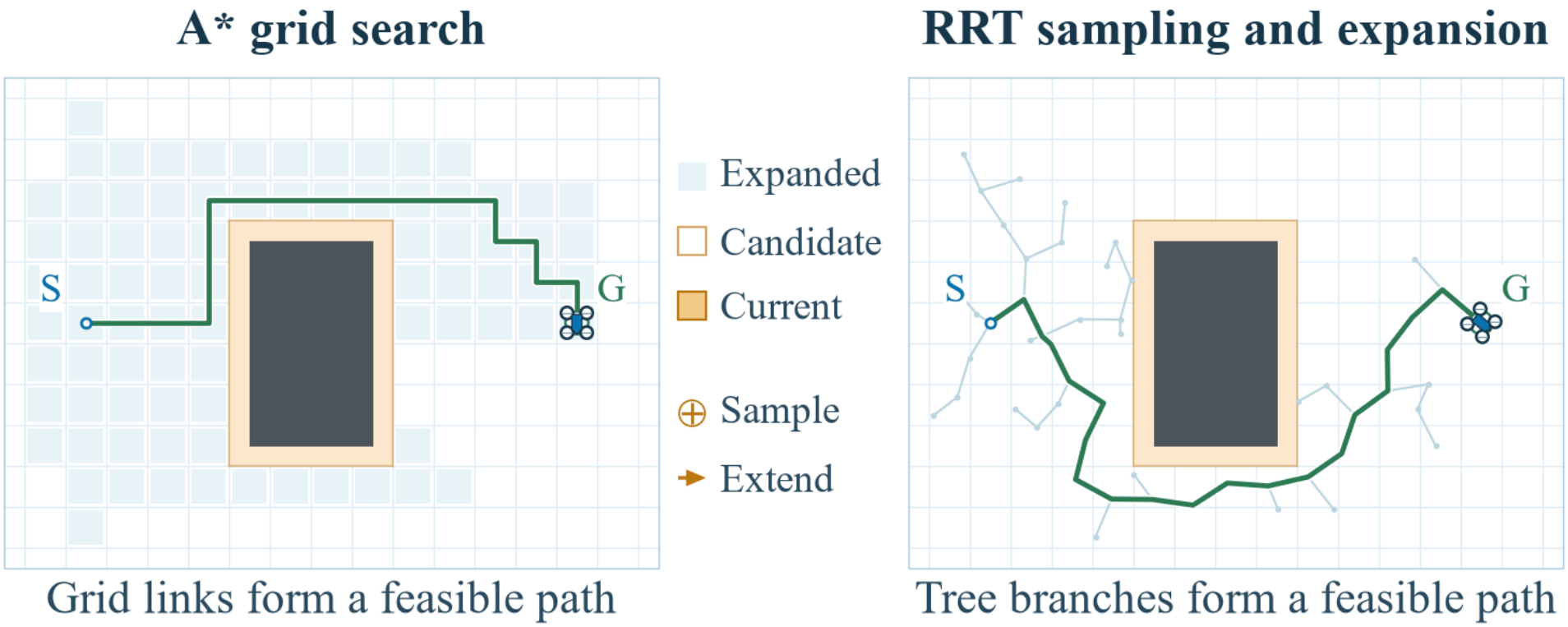


Figure: search selects grid cells by cost; sampling extends to new points. The route needs timing.

6.1.3 Command Sources and Motion References: One-Step Calculation

For the Same 10 m Segment, Travel Time Determines Speed

A 10 m constant-speed segment from 0 to 10 m; time starts on entry and excludes starting and stopping.

Segment Time	Constant Speed	Position at 5 s
20 s	$v = 10/20 = 0.5\ \mathrm{m/s}$	$x(5) = 0.5 \times 5 = 2.5\ \mathrm{m}$
10 s	$v = 10/10 = 1\ \mathrm{m/s}$	$x(5) = 1 \times 5 = 5\ \mathrm{m}$

v is speed and x is the reference position along the line; the path is the same, but timing differs. If the reference takes 10 s and actual motion takes 20 s, the error at 5 s is 5−2.5=2.5 m.

6.1.3 Command Sources and Motion References: Start–Stop Trajectory

Start from Rest and Stop Smoothly at the Target

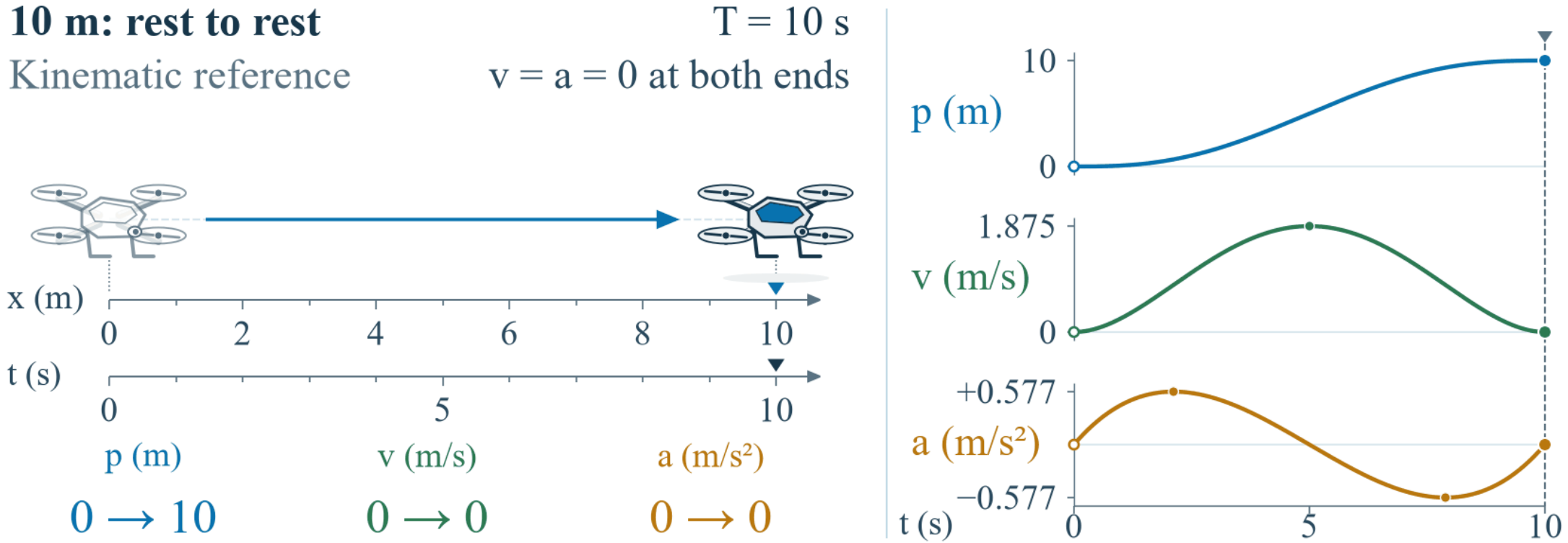


Figure: vehicle position, velocity, and acceleration for one 10 m start–stop reference

6.1.3 Command Sources and Motion References: Time Allocation

How Much Does Slowing a Trajectory Reduce Motion Demands?

Keep the 10 m start–stop path and normalized motion profile; only extend its duration from 10 s to 20 s.

Completion Time	Peak Speed	Peak Acceleration
10 s	$1.875\ \mathrm{m/s}$	$0.577\ \mathrm{m/s^2}$
20 s	$0.9375\ \mathrm{m/s}$	$0.144\ \mathrm{m/s^2}$

$$v_{\max} \propto T^{-1} \qquad a_{\max} \propto T^{-2}$$

Double the time: peak speed halves and acceleration falls to one quarter; path length remains 10 m.

Chapter Outline

External Control and Trajectory Planning: Chapter Roadmap

Chapter 6
External Control and Trajectory Planning

6.1 External Control Fundamentals
6.2 Control Principles and Modes
6.3 Trajectory Models and Generation
6.4 Planning and Obstacle Avoidance
6.5 Tracking and Aggressive Flight Control
6.6 Frontier Control Technologies
6.7 Platform and Communication Interfaces
6.8 Examples and Experimental Validation
6.9 Exercises and Classroom Discussion

4 Units in This Section

6.1.1 Background and Significance
Textbook: Sec. 6.1.1.1

6.1.2 Control Layering
Textbook: Sec. 6.1.1.2

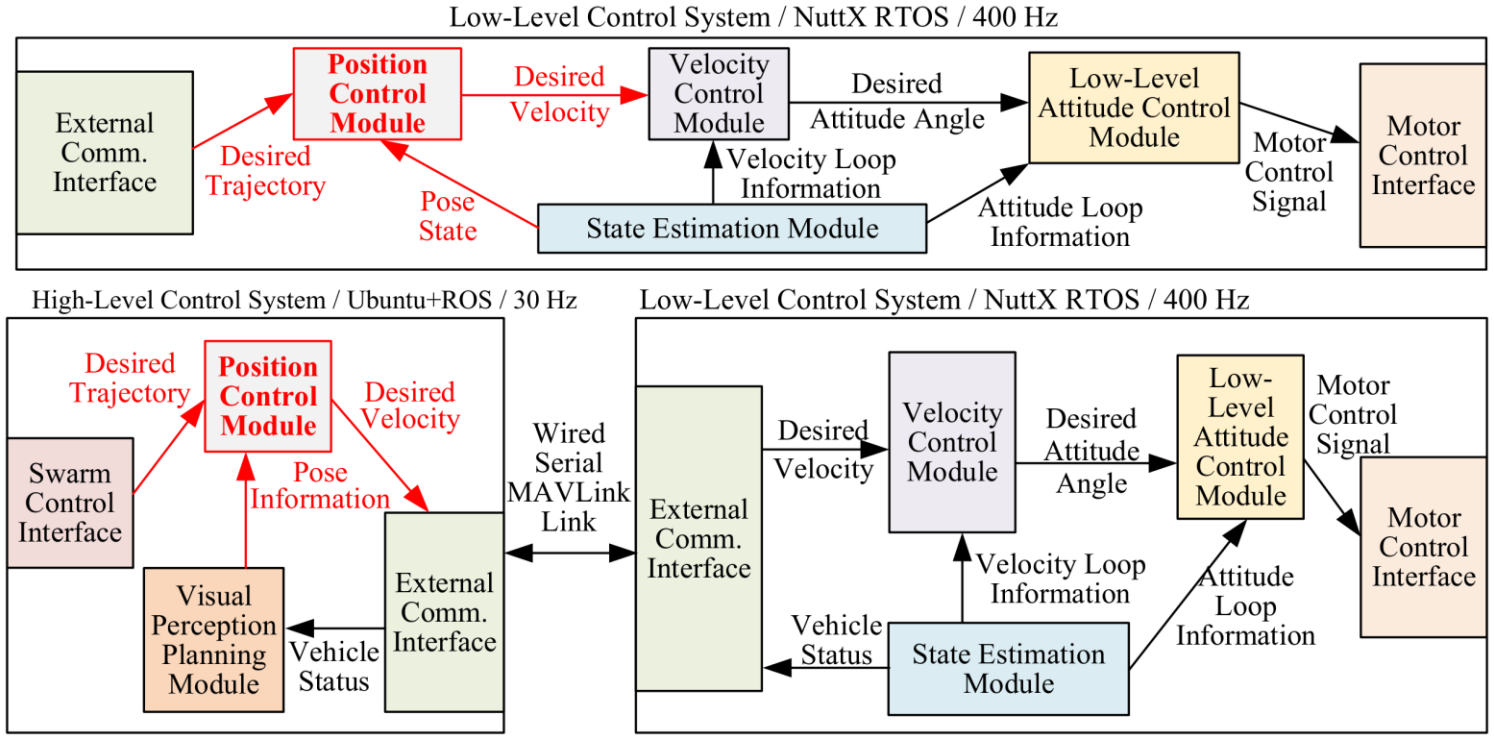
6.1.3 Command Sources and Motion References
Textbook: Secs. 6.1.1.3 and 6.1.3

6.1.4 Performance Analysis
Textbook: Sec. 6.1.1.4

6.1.4 Performance Analysis: Internal and External Operation

The Same Position Controller Can Run on Either Side of the Boundary

Figure 6.2 Internal and External Operation of Trajectory Tracking Algorithms

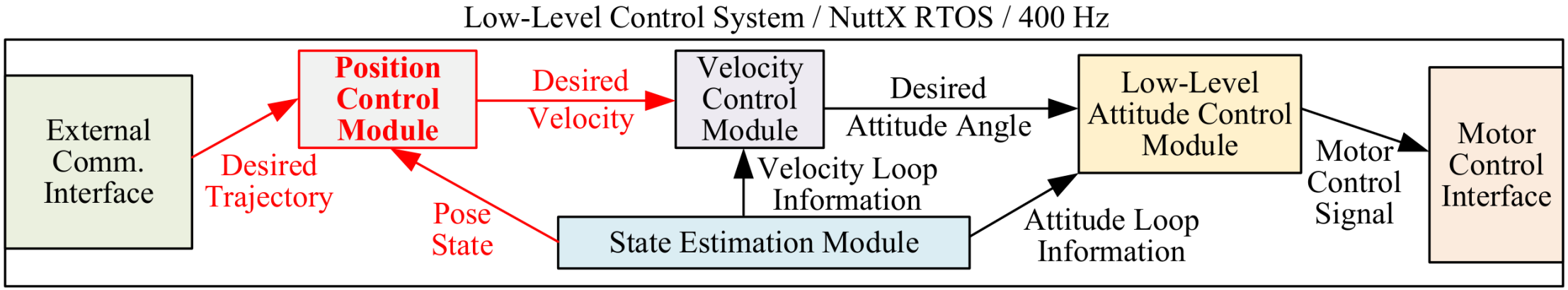


NuttX: an autopilot real-time operating system; ROS (Robot Operating System): a module communication framework. 30 Hz and 400 Hz are illustrative.

6.1.4 Performance Analysis: Internal Operation

Internal Operation: Close Position Feedback Within the Low-Level System

Figure 6.2 (top) Internal position control



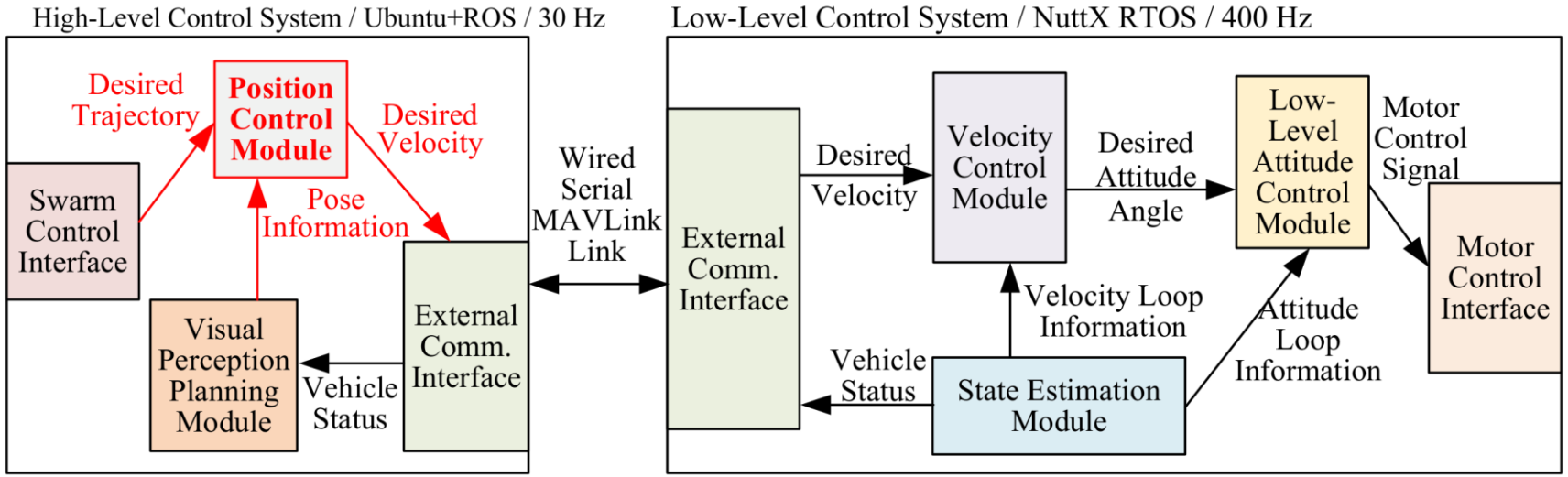
Input: an external module provides the desired trajectory; the internal position controller computes position error.

Feedback: filtering and estimation provide position and other states; internal control loops continue driving the actuators.

6.1.4 Performance Analysis: External Operation

External Operation: Send States Out and Control Targets Back

Figure 6.2 (bottom) External position control



Added feedback stages: state upload → external computation → command download. Evaluate their timing together.

Retained internal control: velocity, attitude, and actuator control remain at the low level; only the position loop moves outside.

6.1.4 Performance Analysis: Feedforward and Feedback

A Moving Target Requires Motion Even When Positions Coincide

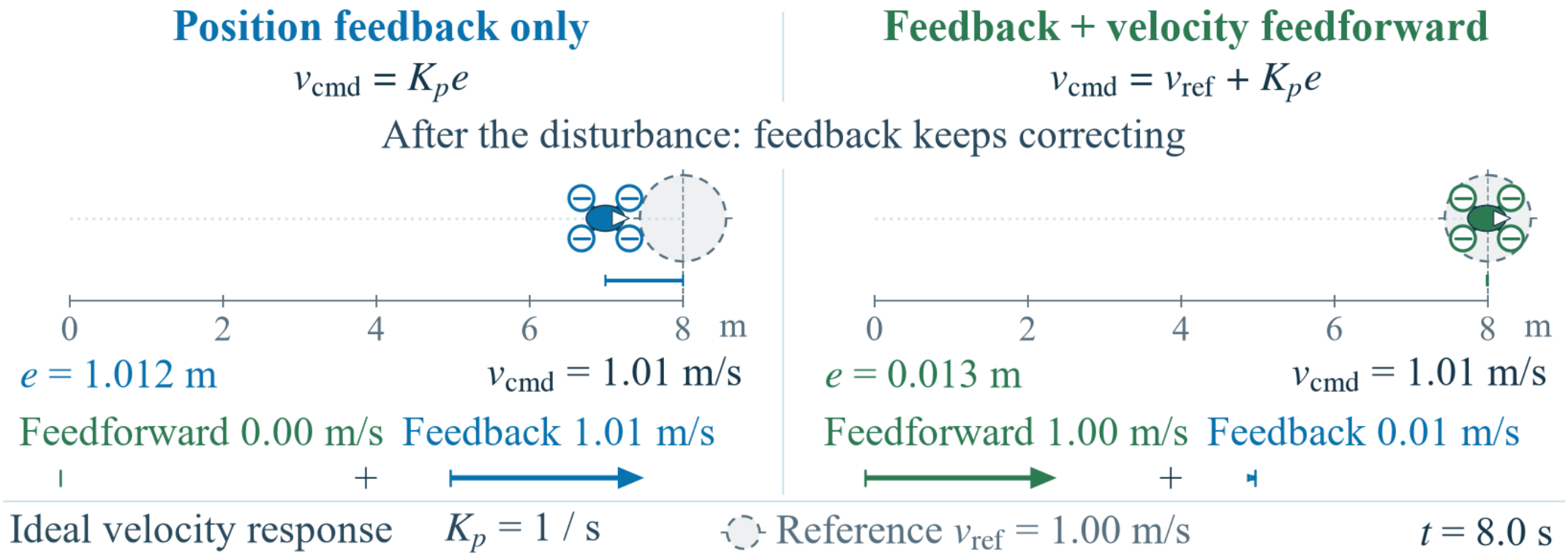


Figure: position feedback alone versus added velocity feedforward under the same velocity disturbance

6.1.4 Performance Analysis: Frequency and Latency

Update Rate Measures How Often; Latency Measures How Late

Example: send at 50 Hz; a cycle from state sampling to command effect takes 100 ms.

$$T_{\text{send}} = \frac{1}{50} = 0.02\,\text{s} = 20\,\text{ms}$$

Send period: the interval between successive send actions.

$$\tau = 100\,\text{ms} = 5T_{\text{send}}$$

End-to-end latency: sampling-to-effect time for the corresponding control action.

Latency includes scheduling, computation, transmission, and execution after sampling; examine variation as well as the mean.

Illustrative rates: internal attitude loops on a real-time operating system often run at 250–1000 Hz; external programs often run at 10–50 Hz and vary with load. Values depend on implementation.

Faster sending cannot refresh old states; closed-loop control needs timely, consistent data.

6.1.4 Performance Analysis: Maneuver Demands and Execution

Double the Speed, Quadruple the Acceleration Needed to Turn

Uniform circular motion with fixed radius 5 m; compare only the required centripetal acceleration.

$$a_c = \frac{v^2}{R}$$

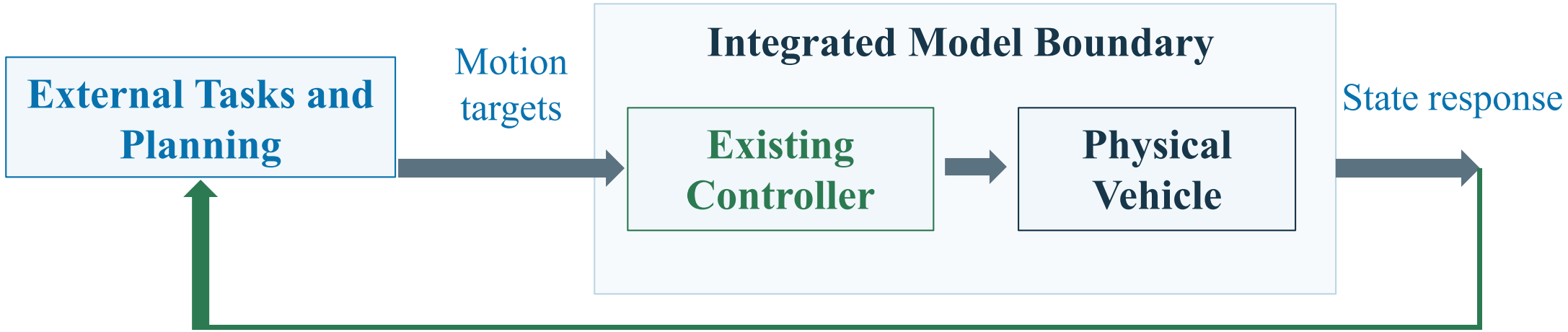
External update rate and latency can readily limit fast, high-acceleration tracking.

Speed	Centripetal Acceleration	Reference Travel During Latency	Relative Motion Demands
1 m/s	$a_c = 1^2/5 = 0.2\ \mathrm{m/s^2}$	0.1 m	Baseline
2 m/s	$a_c = 2^2/5 = 0.8\ \mathrm{m/s^2}$	0.2 m	4× acceleration, 2× advance

v is speed, R radius, and a_c centripetal acceleration; $v\tau$ estimates reference travel during latency τ=100 ms at constant speed.

6.1.4 Performance Analysis: Development and Model Simplification

An Integrated Model Describes Controller and Vehicle Response Together



Continuous motion: replace the vehicle with a low-order model, such as a point mass, to accelerate simulation and training in Simulink or Python.

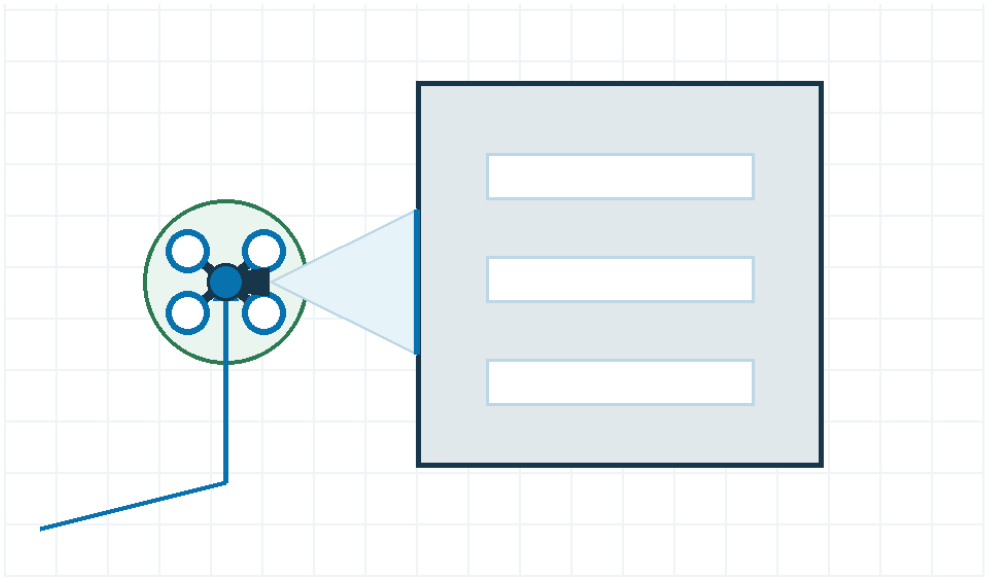
Discrete tasks: use a state machine (condition-based mode transitions) to organize takeoff, inspection, and return.

Model limits: response is delayed, tracking speed is finite, and outputs are bounded; the model applies only within a limited range.

6.1.4 Performance Analysis: Task States and Completion Conditions

After Sending a Target, How Do We Know Inspection Is Complete?

Stationary inspection: reach the site, settle, take an image, then move to the next point.



Arrival

Position error is within tolerance; feedback is fresh and the control mode is valid.

Stable Dwell

Hold these conditions for the specified duration; restart timing on leaving the region.

Image Confirmation

After triggering the camera, confirm capture or saving before moving to the next point.

Entering the region still requires settling and imaging.

6.7 example: error ≤0.20 m for 2 continuous seconds; adapt these thresholds to the task.

6.1.4 Performance Analysis: Observing Model Results

The Same Reference Circle Produces Different Model Responses

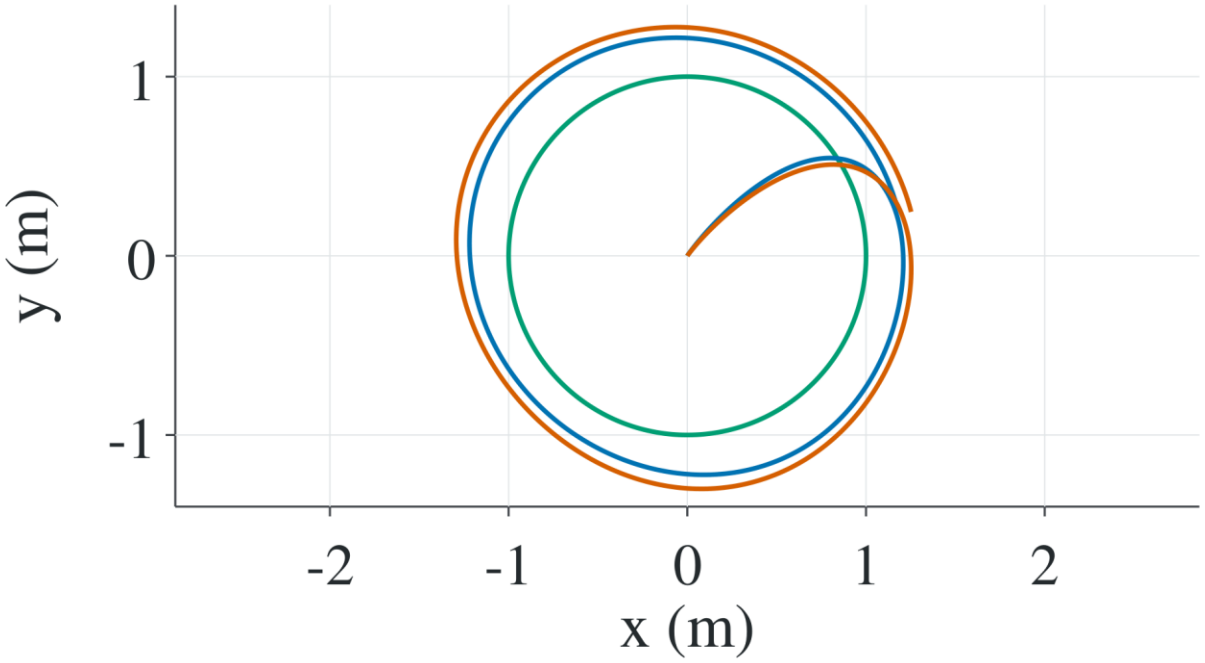


Figure: circular-trajectory model results (first 1200 samples, with added green reference)

Three Curves (T=10 s Circle)

- Reference input
- Identified transfer-function model (sim1.0)
- Nonlinear physics-based model (sim2.0)

Close outputs: similar responses here.

Shape: spatial responses differ from the reference.

Radius: reference 1 m; output maxima about 1.25 m and 1.32 m.

Align times for simultaneous error.

6.1.4 Performance Analysis: Running the Example

Reproduce the Circular-Trajectory Model Comparison in MATLAB

Directory: [installation directory]/RflySimAPIs/6.RflySimExtCtrl/1.BasicExps/e9_TrajectoireFollowing_Segment/e3.1/sim2.0/

Entry: Plot_Compare_cercle.m; see e3.1/Readme.pdf, Sec. 4.2.

Steps: set MATLAB's current folder to sim2.0 → run the script → check both output curves and legends.

Missing data? Confirm current folder sim2.0 contains:
N4_Step1_cercle_comparaison_T10.mat
N4_Step1_cercle_comparaison_T10tf.mat

Expected: black dotted—nonlinear; blue—TF.
One plot: Output Cercle T=10s Model/TF.
Reference: green target, other colors; use legends.

Regenerate (optional): in sim1.0 and sim2.0, simulate the T=10 s circle and run Save_Cercle.m in each. Copy the new ⋯T10tf.mat from sim1.0 to sim2.0. See the four steps in 6.8.2, "Saving and Comparing."

6.1.4 Performance Analysis: Recording and Validation

Time-Aligned Data Are Needed to Evaluate Tracking

The circle plot retains only spatial shape. To check on-time tracking, align reference and estimated states to the same time.

Example of time-aligned northward motion records (same frame; positions in m)

Time / s	Reference Position	Estimated Position	Position Error
2.0	2.00	1.80	0.20
3.0	3.00	2.75	0.25
4.0	4.00	3.90	0.10

Also record control requests, mode, and task state to link errors to execution.

This illustrates a record format; the example's old MAT files lack an explicit time column and cannot directly provide simultaneous error.

6.1.4 Performance Analysis: Engineering Tradeoffs

Choose Deployment by Performance and Development Cost

Consideration	Internal Operation	External Operation
Interfaces and timing	Shorter signal paths simplify coordinated scheduling	Read states and send commands; evaluate latency
Development and resources	Embedded resources and firmware workflows impose limits	Easier integration of perception, planning, and learning software
Models and validation	Focus on control loops and actuator dynamics	Focus on interface response and task constraints

Meet task timing and motion requirements first, then choose the control boundary and tools.

Classroom Exercise

Check Control Responsibilities and Timing With Five Questions

Question	Key Points
Is an onboard program always internal control?	No. Judge its boundary relative to the low-level system.
Does staying on the reference circle mean accurate tracking?	Not necessarily. Compare reference and feedback at the same time.
Does sending at 50 Hz imply 20 ms latency?	No. 20 ms is the send period; measure latency separately.
Is a three-layer control system necessarily a distributed swarm?	No. The three-layer example is distributed; layering and decision organization are separate dimensions.
Send rate 20 Hz, latency 150 ms: what is the send period, and how many periods is the latency?	Period: 1/20 s=50 ms; 150 ms equals 3 send periods.

Classroom Exercise

Calculate Acceleration, Travel During Latency, and Tracking Error

Question	Key Points
Circle radius 5 m, constant speed 3 m/s: what centripetal acceleration is required?	$a_c = v^2/R = 3^2/5 = 1.8\ \mathrm{m/s^2}$, 9 times the value at 1 m/s.
At the same speed with 150 ms end-to-end latency, how far does the reference advance? (Constant-speed estimate)	$v\tau = 3 \times 0.15 = 0.45\ \mathrm{m}$, 3 times the value at 1 m/s; higher speed increases the effect of old states.
Enter a 12 m straight segment together: reference takes 6 s, actual motion 8 s, both at constant speed. Find simultaneous error and distance to the path at 4 s.	Reference: 2 m/s×4 s=8 m; actual: 1.5 m/s×4 s=6 m. Simultaneous error: 2 m; distance to path: 0 m.

Section Summary

Connect Task Requirements to an Observable, Testable Motion Loop

Responsibilities: tasks and planning set motion requirements; low-level control produces the response; state feedback enables continual correction.

Boundaries: functional layers are distinct from hardware location; external control can also run on an onboard computer.

Motion: a path defines the spatial route; a trajectory adds arrival times. Execution also depends on vehicle capabilities.

Performance: measure send period, end-to-end latency, and tracking error separately; none substitutes for another.

Common errors: treating send period as latency, ignoring time alignment, and equating onboard with low-level.

The same inspection task can use different control modes. Next compare the roles of people and programs.