UAV Flight Log Generation and High-Level Mission Planning Using Vision-Language Models

Author: Tsung-Yen, Yu

Advisor: Dr. Wen-Hung Liao

content

Abstract & Background

- The rapid rise of AI in computer vision is accelerating the future of autonomous systems
- For drones, this opens the door to move beyond simple GPS routes to true situational awareness
- Simultaneously, advances in NLP let us "talk" to AI in new ways

The Goal: To merge these fields, enabling a human to effortlessly collaborate with a drone

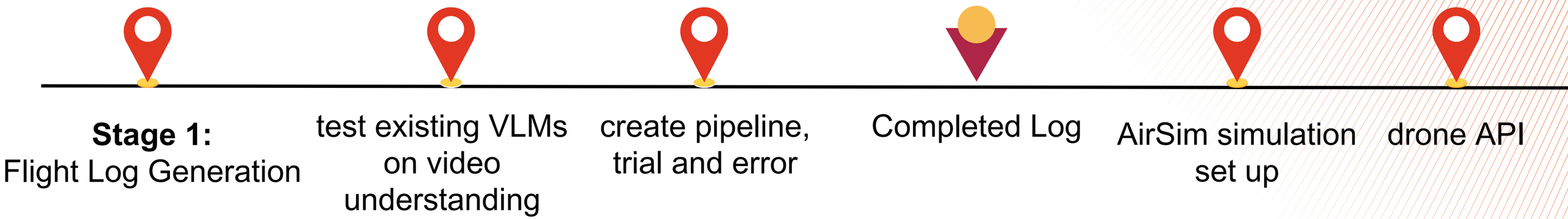
Abstract & Background

- The rapid rise of AI in computer vision is accelerating the future of autonomous systems
- For drones, this opens the door to move beyond simple GPS routes to true situational awareness
- Simultaneously, advances in NLP let us "talk" to AI in new ways

The Goal: To merge these fields, enabling a human to effortlessly collaborate with a drone

1. Integrate Vision-Language Models (VLMs) with drone control via multi-layer pipeline
2. Use a logbook-like approach to document the flight details

Methods - Overview



Stage 1: Flight Log Generation

Methods - VLM Selection

To test existing VLMs, I first tested their ability to summarize a few consecutive frames.

- LLaVA (13b):
  - Run on local machine, fastest and cost-free
  - NLP backbone is too weak to reliably solve complex vision tasks
- ChatGpt:
  - Requires API key access
  - Richest contextual descriptions at a moderate speed
- Claude:
  - Requires API key access
  - Accurate contextual descriptions at a high speed
  - Accepts the longest context window
- Gemini:
  - Requires API key access
  - Accurate contextual descriptions at a moderate speed
  - Provides video understanding capabilities

Methods - VLM Selection

To test existing VLMs, I first tested their ability to summarize a few consecutive frames.

- LLaVA (13b):
  - Run on local machine, fastest and cost-free
  - NLP backbone is too weak to reliably solve complex vision tasks
- ChatGpt:
  - Requires API key access
  - Richest contextual descriptions at a moderate speed

← Act as the “eyes”

- Claude:
  - Requires API key access
  - Accurate contextual descriptions at a high speed
  - Accepts the longest context window

← Act as the “brain”

- Gemini:
  - Requires API key access
  - Accurate contextual descriptions at a moderate speed
  - Provides video understanding capabilities

← Act as the “second agent”

Methods - VLM Selection

Base on the vision ability of the VLMs, decide the possible Flight Log Formation:

1. Flight Identification
   - Date
   - Start Time
   - End Time
   - Total Duration
2. Flight Purpose
   - Purpose of Flight
   - Type of Operation
3. Environment Observation
   - Location Name/Description
   - GPS Coordinates (Takeoff)
   - GPS Coordinates (Landing)
   - Weather Conditions:
     - Wind Speed, Wind Direction, Visibility, Temperature, Cloud Cover, Precipitation
4. Flight Parameters
   - Maximum Altitude
   - Maximum Distance
   - Flight Pattern: ☐ Hover ☐ Linear ☐ Orbit ☐ Waypoint ☐ Free Flight
   - Key Waypoints/Locations
   - Flight Path Summary
5. Camera & Recording Settings
   - Video Resolution
   - Frame Rate
   - Recording Format
6. Safety Consideration
   - Obstacles Present
   - People in Area
   - Emergency Landing Sites
7. Incidents
   - Any Issues Encountered
   - Wildlife Interactions
   - Signal Loss Events
   - Weather Changes
   - Equipment Malfunctions
8. Notes & Lessons Learned
   - Flight Performance
   - Footage Quality
   - Areas for Improvement
   - Future Considerations

test existing VLMs on video understanding

create pipeline, trial and error

Completed Log

Methods - Pipeline Design

Utilize the **Wolf** framework created by NVIDIA research team to maintain spacial and temporal information

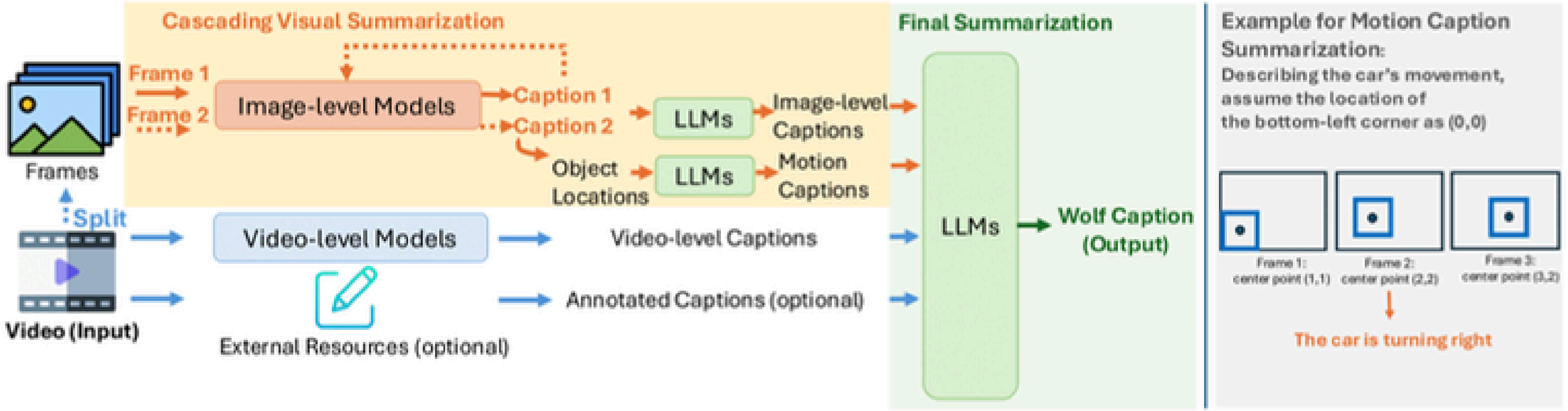


Figure 1. Overview of proposed Wolf framework. Wolf utilizes both image-level and video-level models to generate diverse and detailed captions, which are then summarized for cross-checking. On the right side, we also provide an example of how we obtain motion captions based on object locations extracted from image captions.

Include both the caption of the previous frame and the current frame itself as the final input

Wolf: Dense Video Captioning with a World Summarization Framework

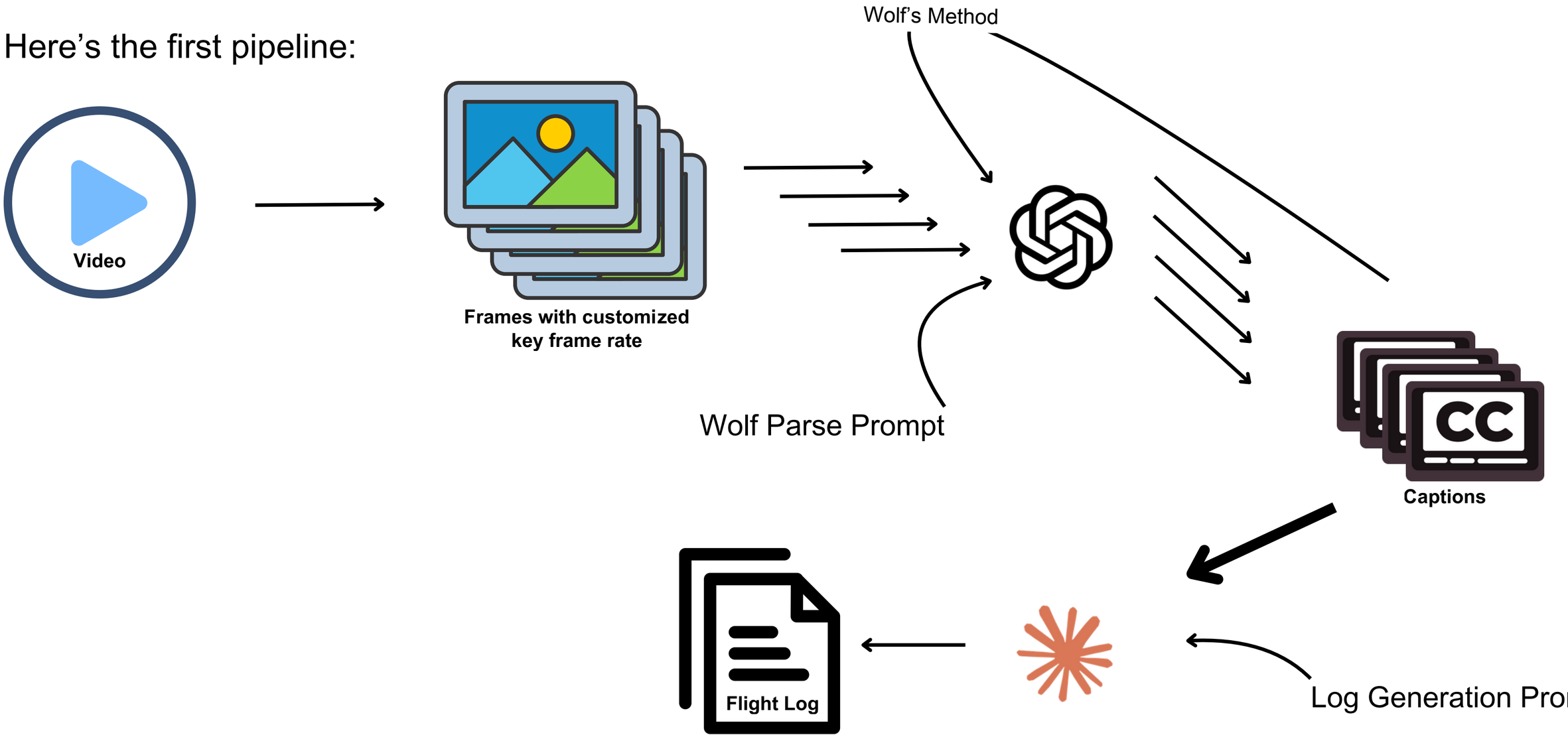
Methods - Pipeline Design

test existing VLMs on video understanding

create pipeline, trial and error

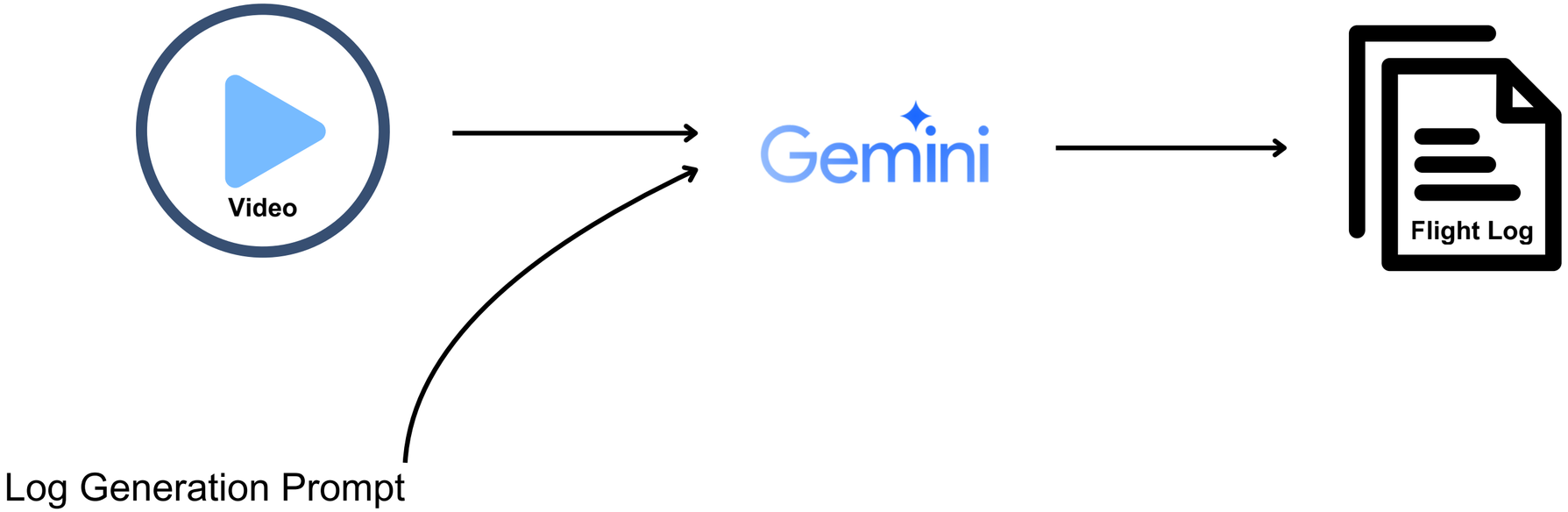
Completed Log

Here’s the first pipeline:



Methods - Pipeline Design

The second pipeline is easier, we just use the Gemini video understanding capabilities that were released in June 2025:



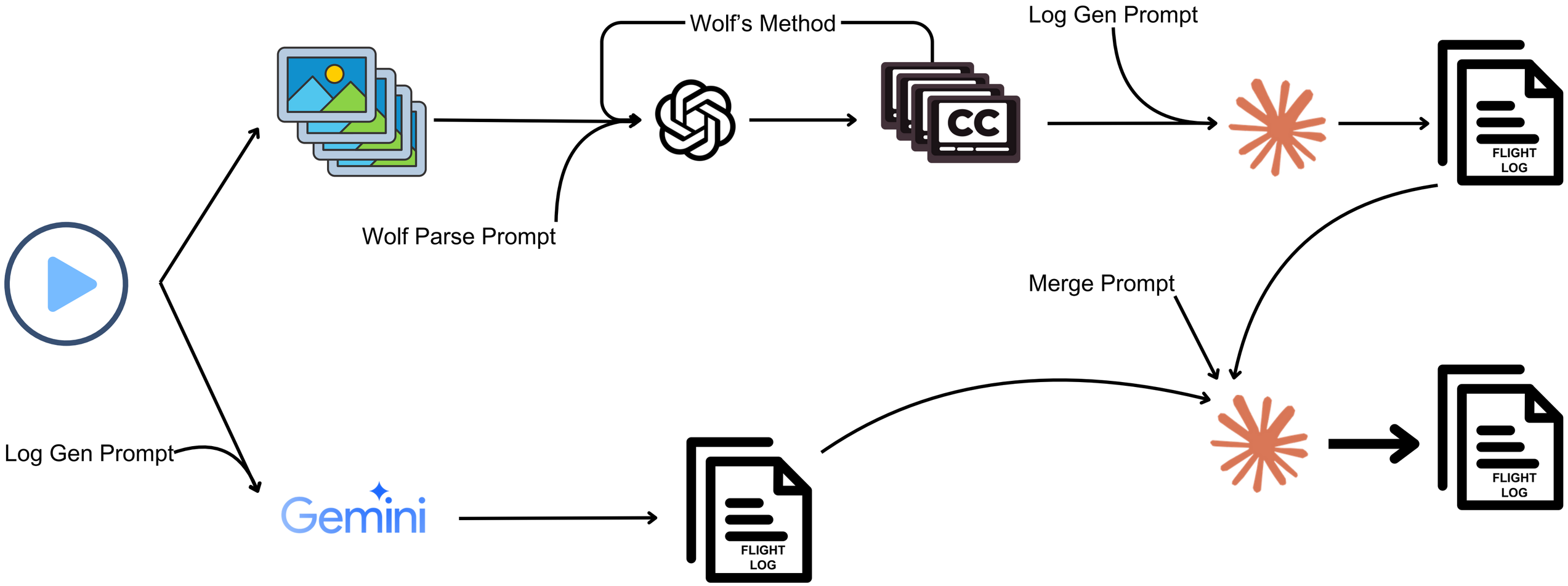
Methods - Pipeline Design

test existing VLMs on video understanding

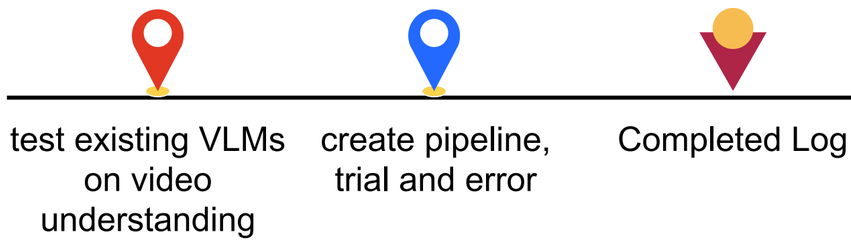
create pipeline, trial and error

Completed Log

Now, we merge the 2 pipelines to complement each other:

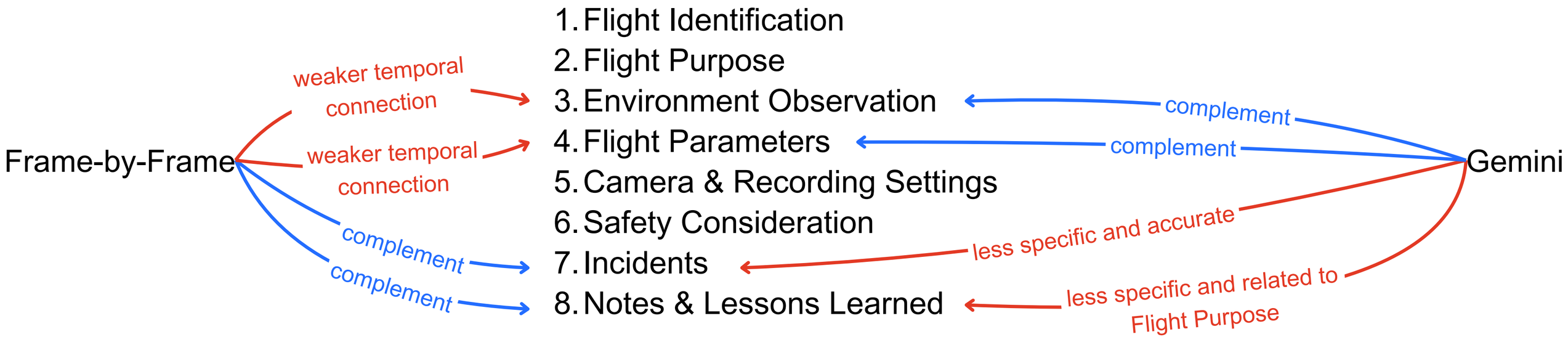


Methods - Pipeline Design



Why use 2 different pipelines?

Let’s call the first method “Frame-by-Frame pipeline”, and the second one “Gemini pipeline”



Methods - Pipeline Design

Outside from “what” the 2 pipelines output, we can also discuss “how long”

- Frame-by-Frame:
  - bottleneck: Internet latency (too many API calls)
  - short video (2 min) → 5 min
  - long video (5 min) → 10+ min
- Gemini:
  - bottleneck: Uploading video to google cloud (drone videos are often 4k)
  - short video (2 min) → 4 min
  - long video (5 min) → 6 min

With multi-threading, Frame-by-Frame pipeline is the bottleneck

Interlude: AirSim Simulator

Methods - Simulation

Why use simulation?

1. Reduce experimental costs
2. Shorten the development cycle
3. Avoid real world network connection overhead

Why AirSim?

1. Built on top of Unreal Engine, suitable for vision tasks
2. Integrated with ROS2, capable of easily adapting to real-world applications when hardware requirements are met

Methods - Drone Basic

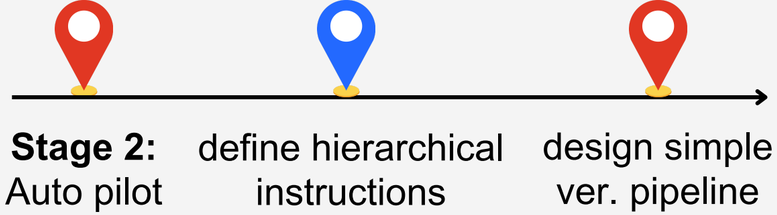
There are multiple ways to control the drone in AirSim, for simplicity, we first allow only:

1. move forward a distance
2. rotate an angle
3. move vertically a height

By utilizing these 3 methods, the drone can reach everywhere

Stage 2: Autonomous Flight

Methods - Define Levels



	Name	Description	Example
High Level	(Vision) Task	Ambiguous, high-level goals without specific "how-to" instructions.	"Find me an old temple", "Fly until you see a roundabout"
High Level	Predefined Mission	Complicated but commonly used. Hard for VLMs to handle	"Return Flight", "Areal Scan"
Middle Level	Navigation	Gives a specific idea of "how" to fly	"Fly ahead for 100m and turn left", "fly in a square that has a length of 100m"
Low Level	Action	The most specific commands (APIs) that directly control the system	move_forward(x), rotate(x)

Methods - Pipeline Design

Goal: Design a pipeline that can seamlessly convert high level and middle level instructions to low level instructions so that we can actually fly the drone in AirSim.

Methods - Pipeline Design

Finished

distributing

Is Predefined

Not Finished

User Instruction

filtering

Not Predefined

decision making

motion planning

extract & execute

decision making

Past Instruction + Actions

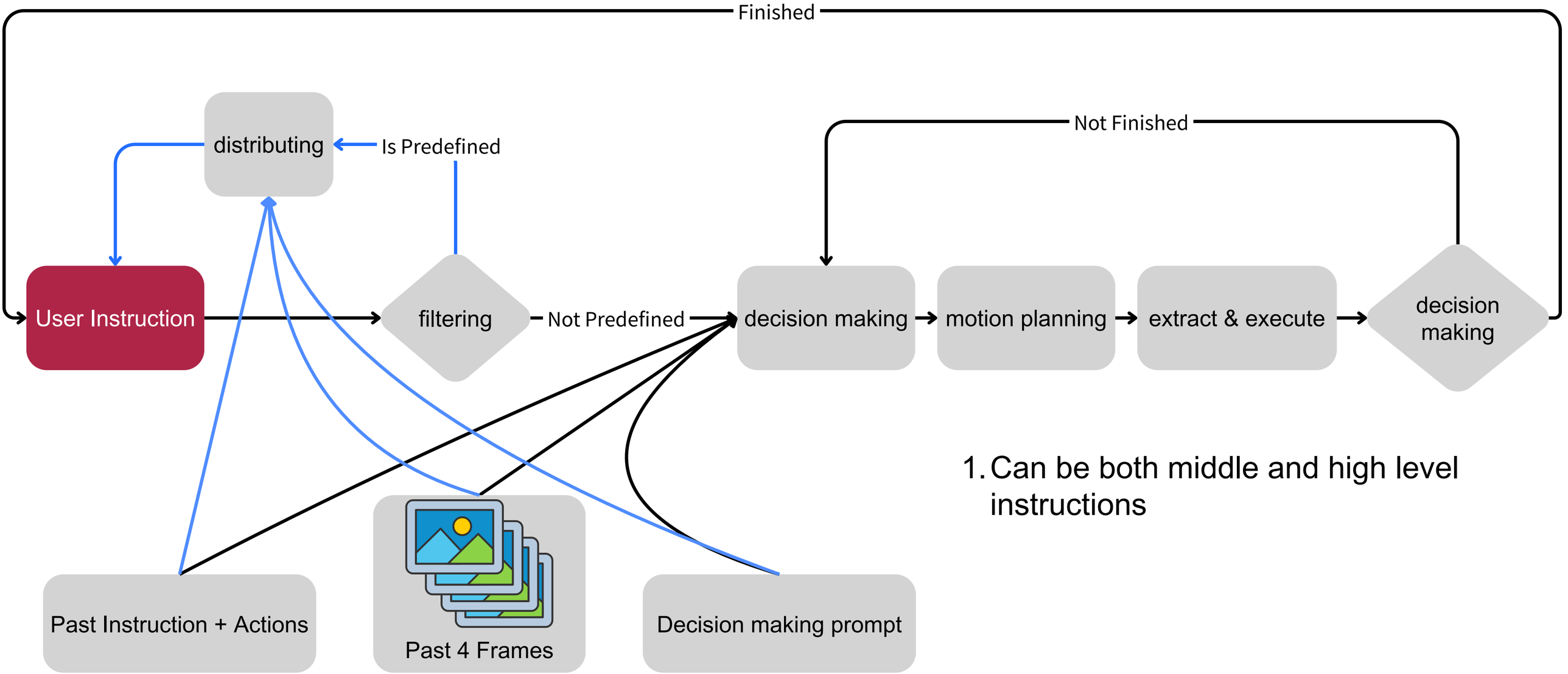
Past 4 Frames

Decision making prompt

output 2 things after the flight

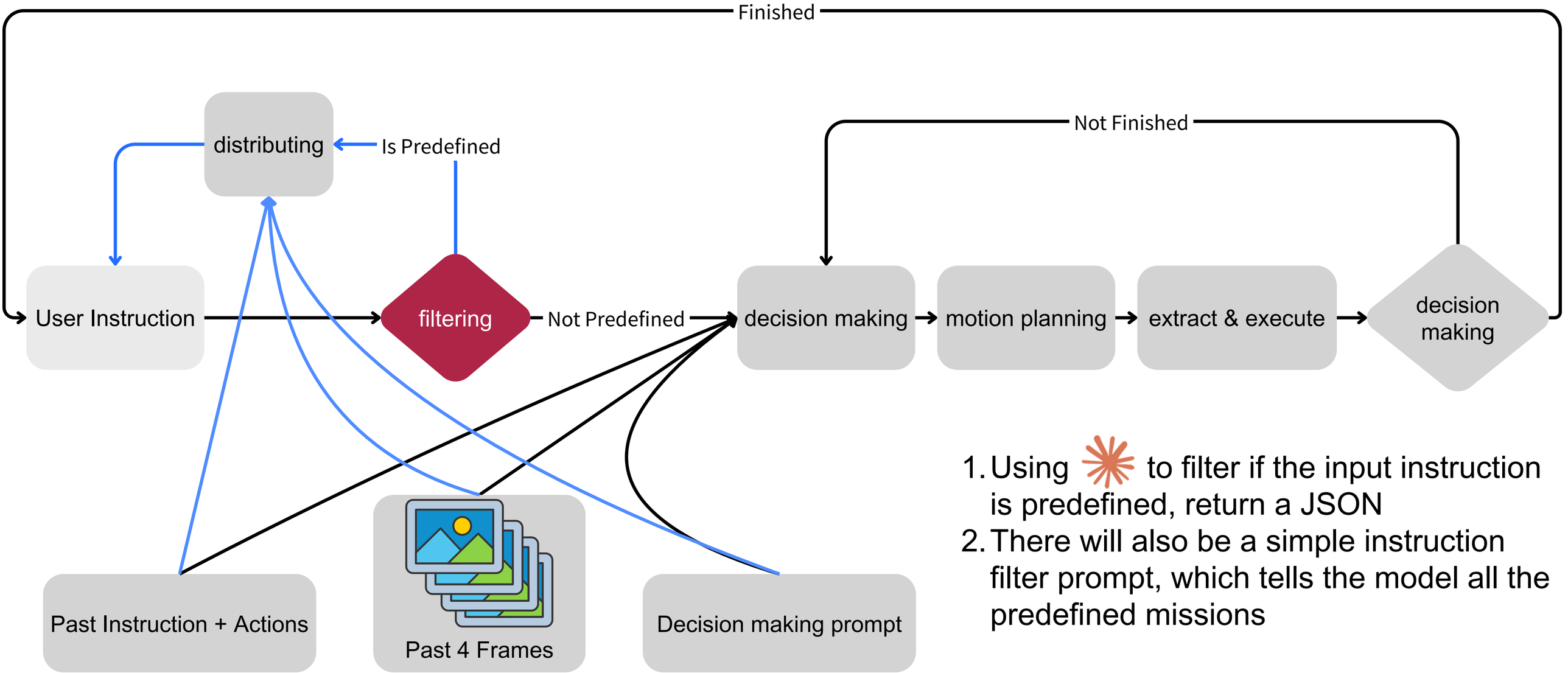
1. Drone footage
2. Action list

Methods - Pipeline Design



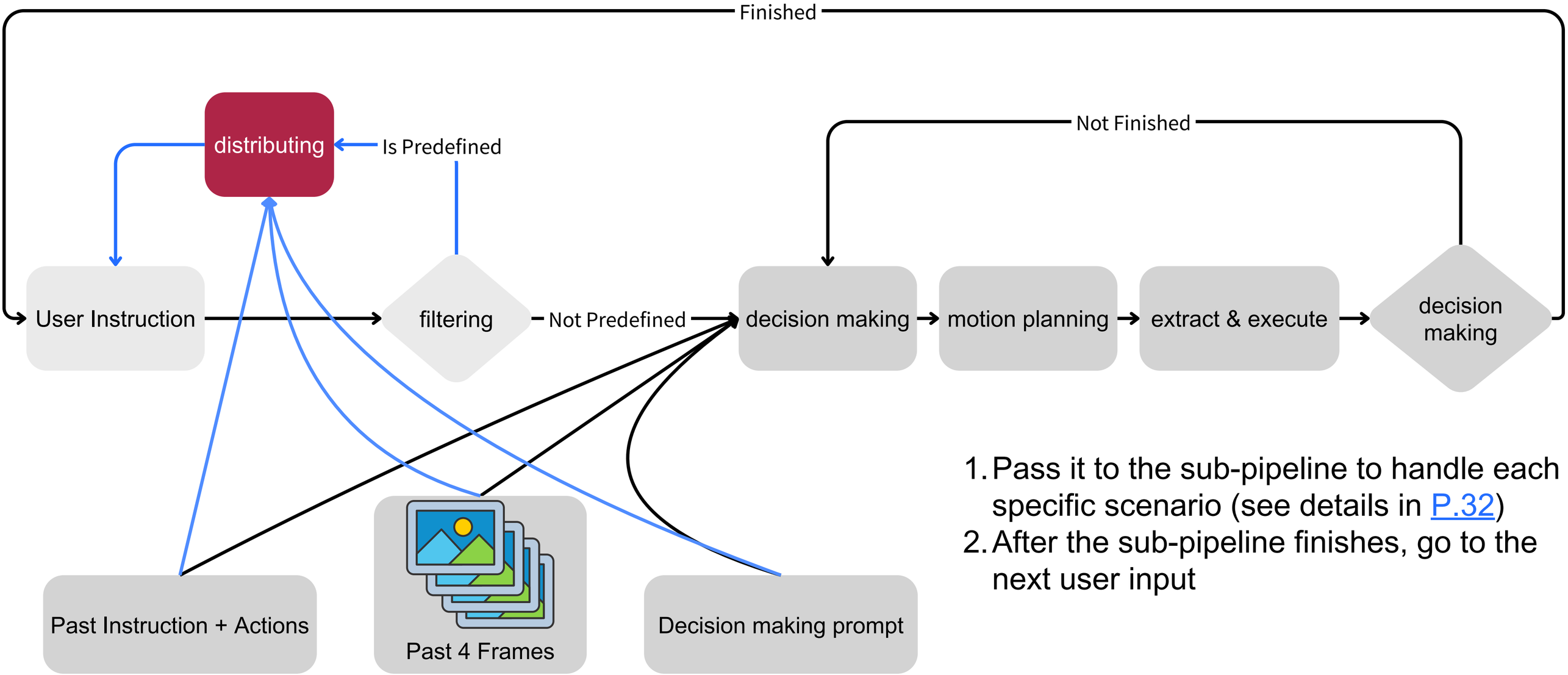
1. Can be both middle and high level instructions

Methods - Pipeline Design



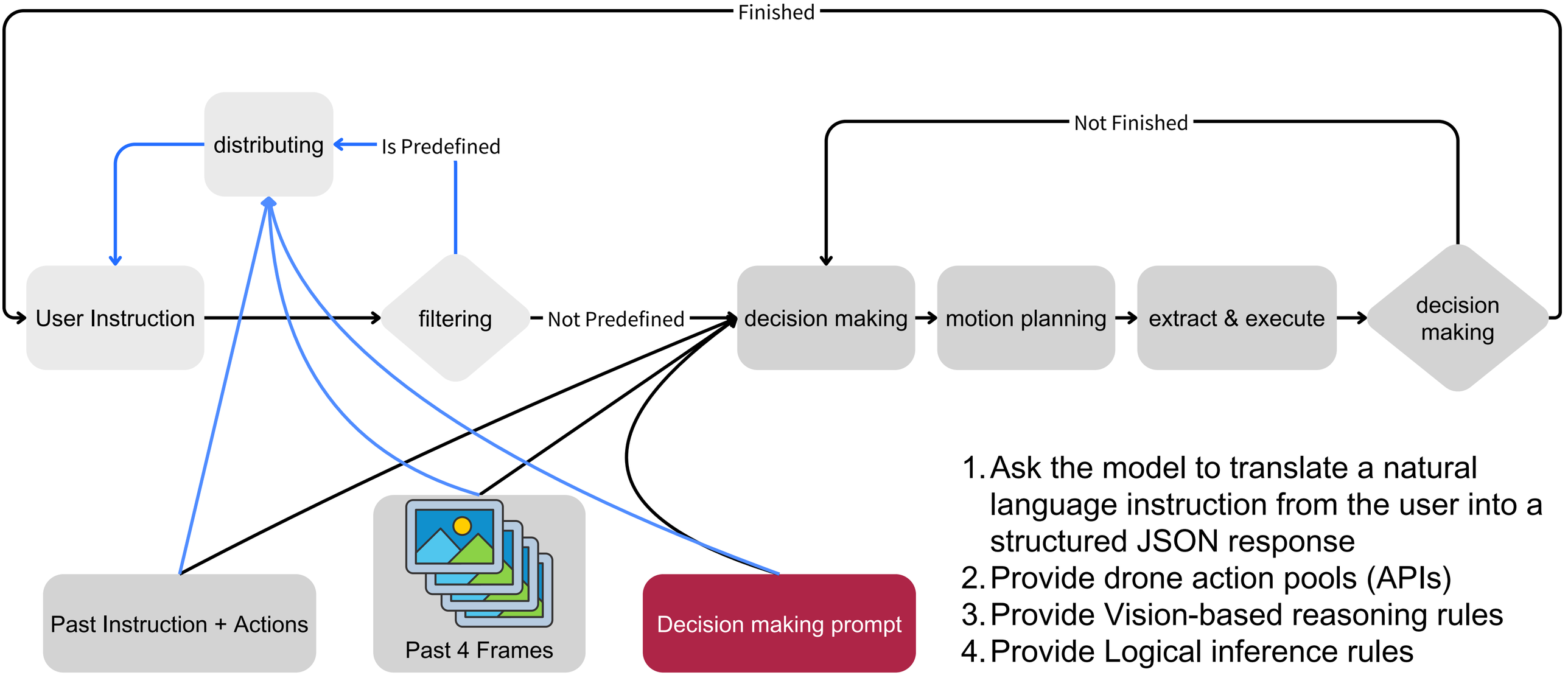
1. Using ✳ to filter if the input instruction is predefined, return a JSON
2. There will also be a simple instruction filter prompt, which tells the model all the predefined missions

Methods - Pipeline Design



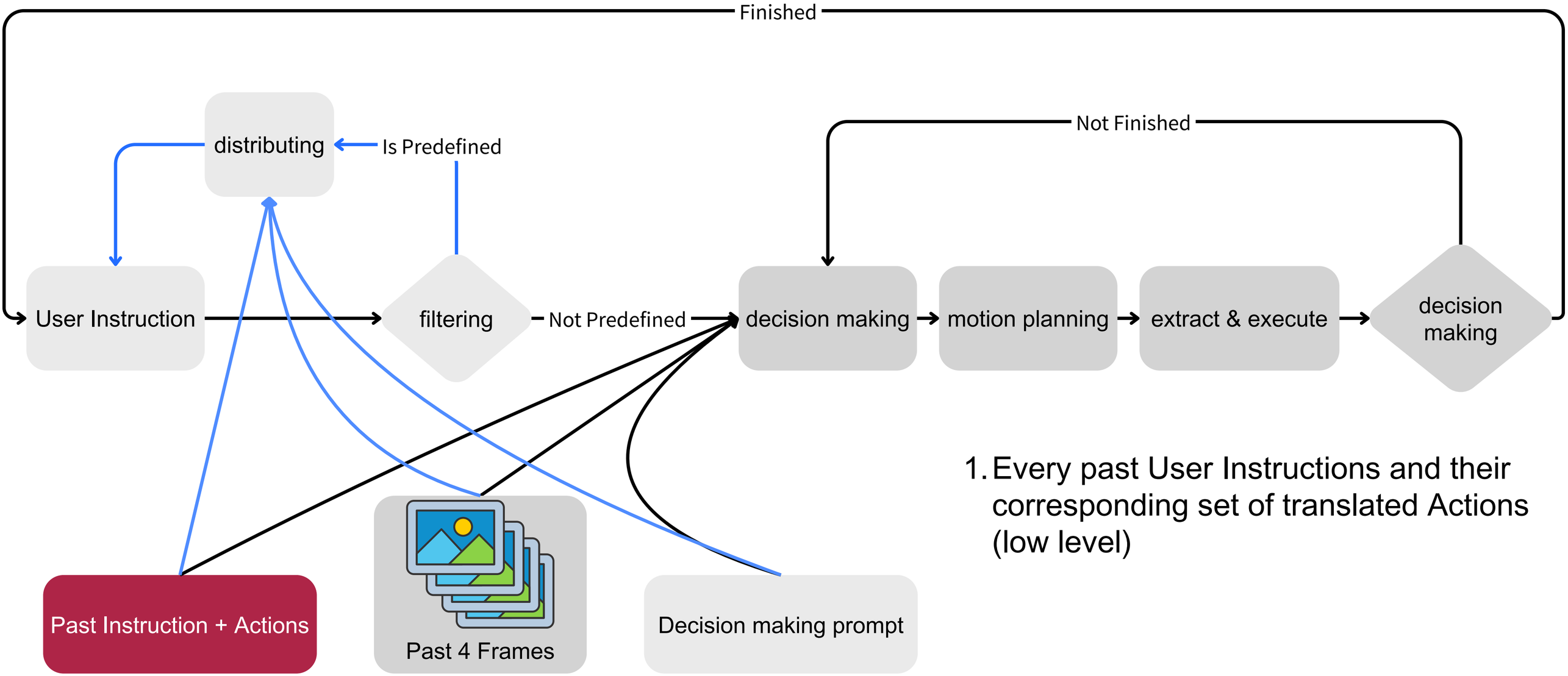
1. Pass it to the sub-pipeline to handle each specific scenario (see details in P.32)
2. After the sub-pipeline finishes, go to the next user input

Methods - Pipeline Design



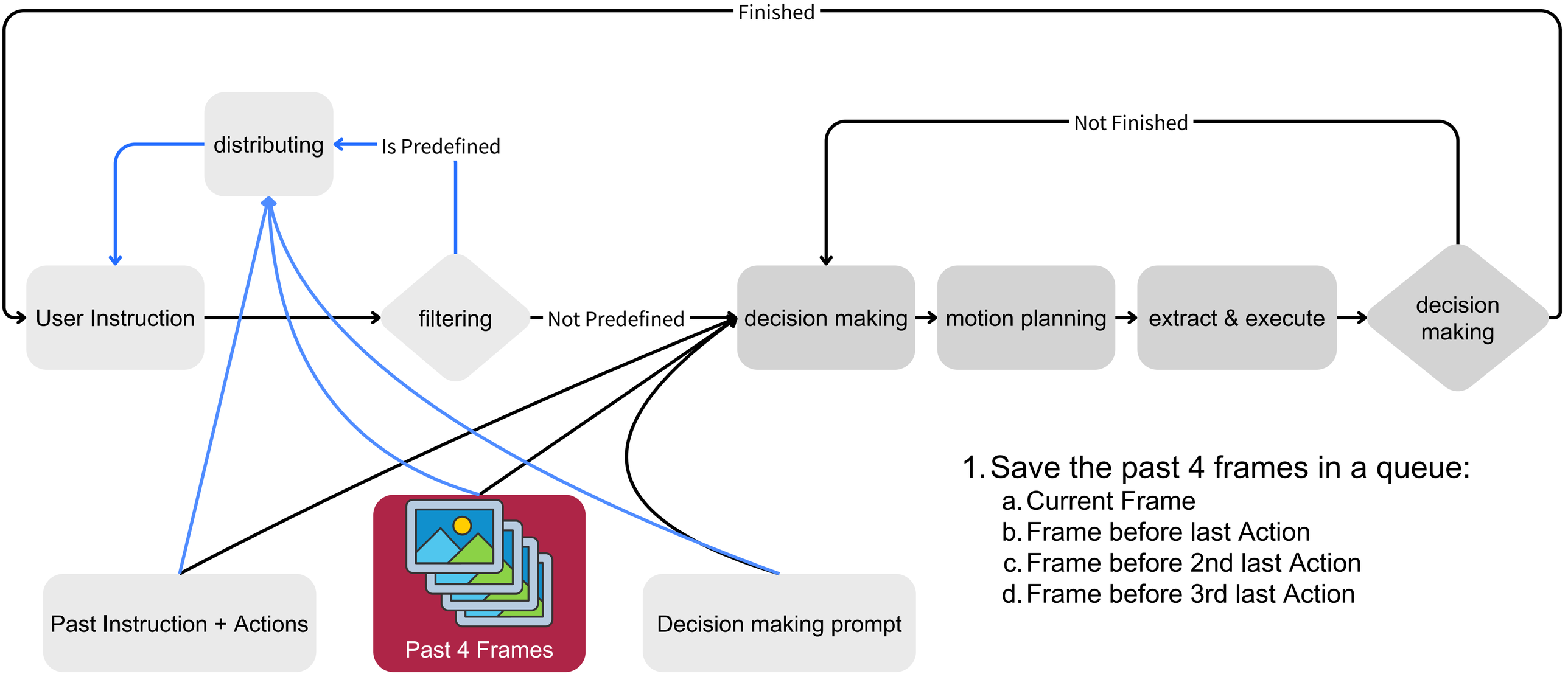
1. Ask the model to translate a natural language instruction from the user into a structured JSON response
2. Provide drone action pools (APIs)
3. Provide Vision-based reasoning rules
4. Provide Logical inference rules

Methods - Pipeline Design



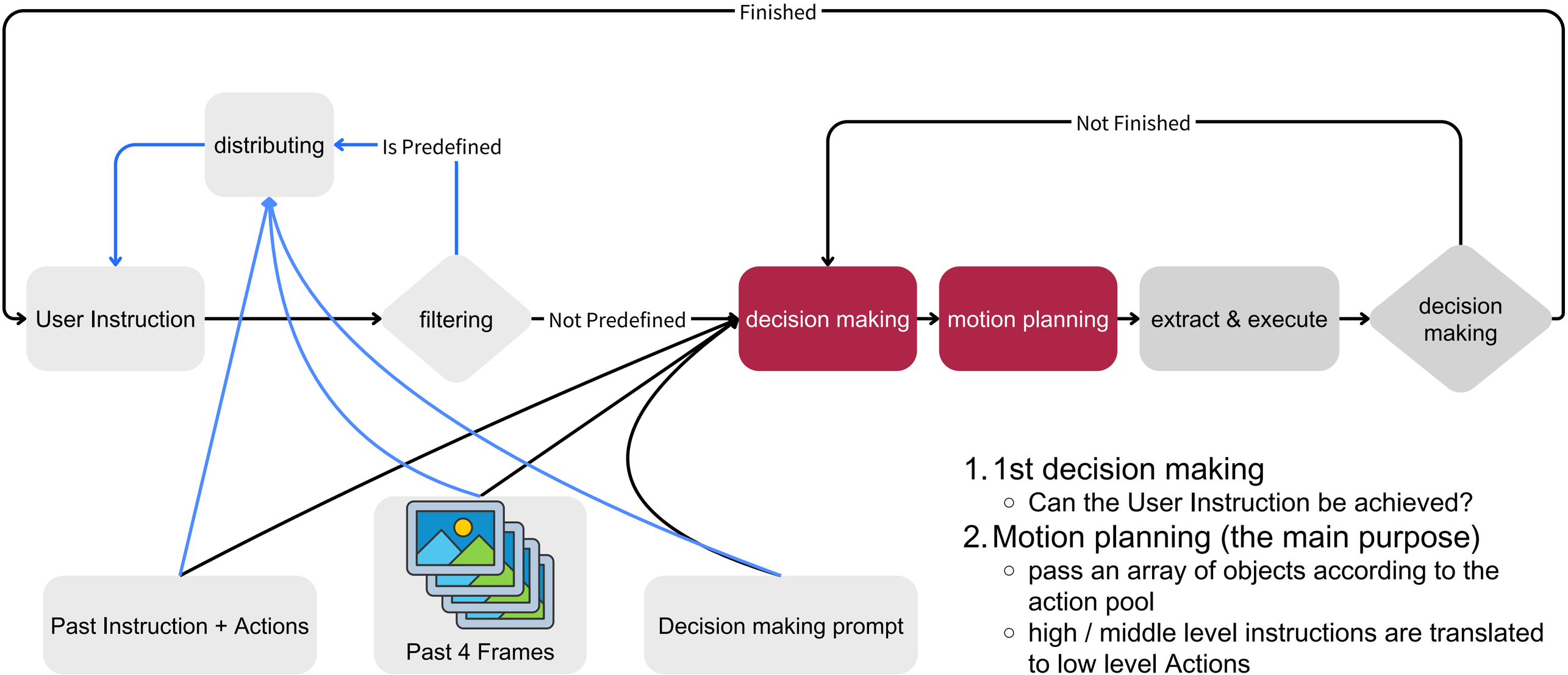
1. Every past User Instructions and their corresponding set of translated Actions (low level)

Methods - Pipeline Design



1. Save the past 4 frames in a queue:
   a. Current Frame
   b. Frame before last Action
   c. Frame before 2nd last Action
   d. Frame before 3rd last Action

Methods - Pipeline Design



1. 1st decision making
   - Can the User Instruction be achieved?
2. Motion planning (the main purpose)
   - pass an array of objects according to the action pool
   - high / middle level instructions are translated to low level Actions

Methods - Pipeline Design

Finished

distributing

Is Predefined

Not Finished

User Instruction

filtering

Not Predefined

decision making

motion planning

extract & execute

decision making

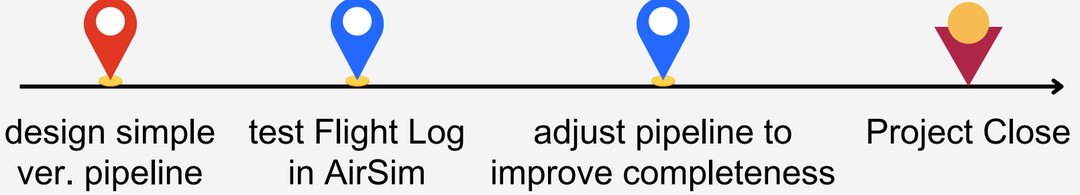
Past Instruction + Actions

Past 4 Frames

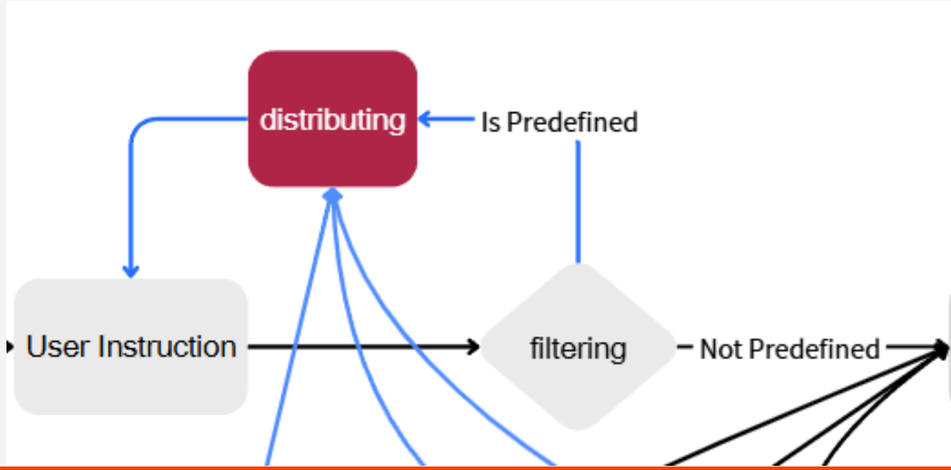
Decision making prompt

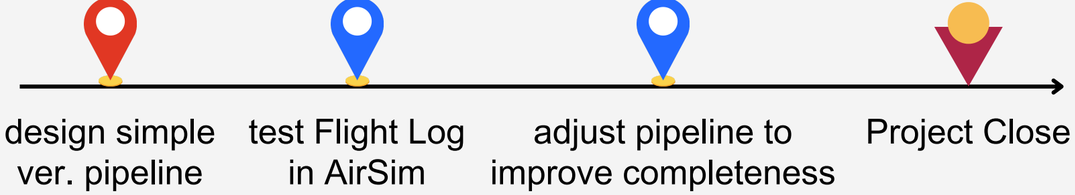
1. Start flying in AirSim
2. After all Actions are executed, do the 2nd decision making
   - Is the User Instruction satisfied?

Methods - Adjustment



1. Try out what Flight Logs will look like in the AirSim simulation environment (see detail in the Result tab at P.67)
2. Start customizing sub-pipelines for predefined missions





Methods - return flight

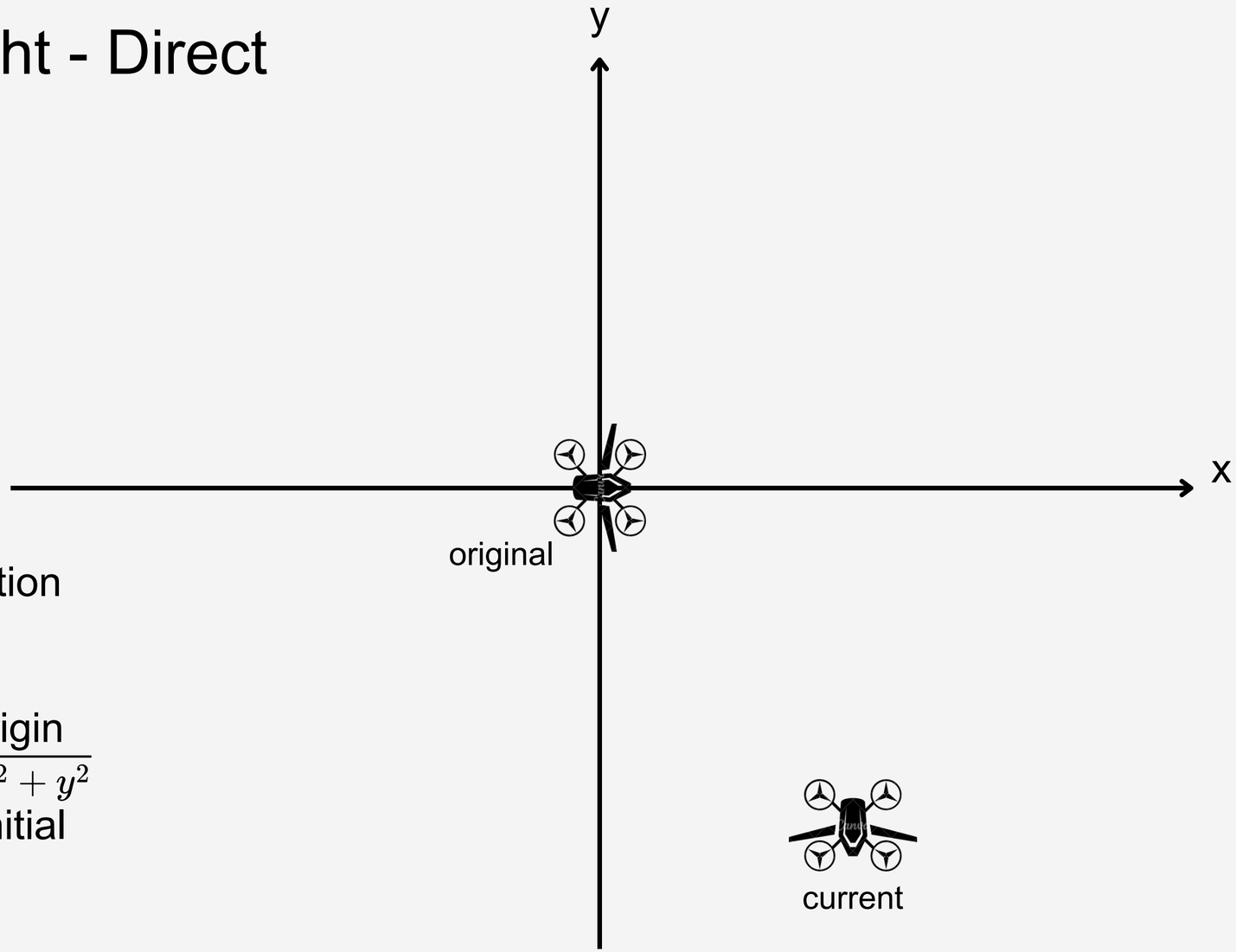
Return Flight - Direct

What we need:

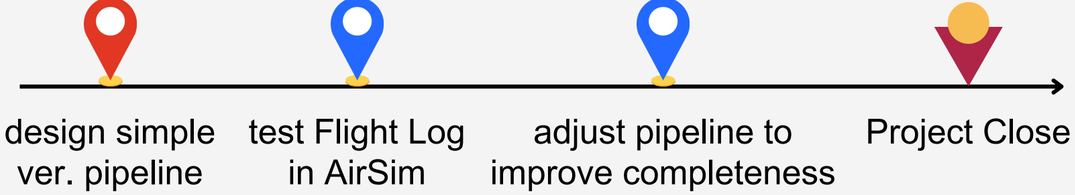
1. Past Actions

Idea (Big picture):

1. get the current position (x,y)
2. get net rotation
3. rotate to face the origin
4. fly a distance of $\sqrt{x^2 + y^2}$
5. rotate back to the initial facing direction



Methods - return flight

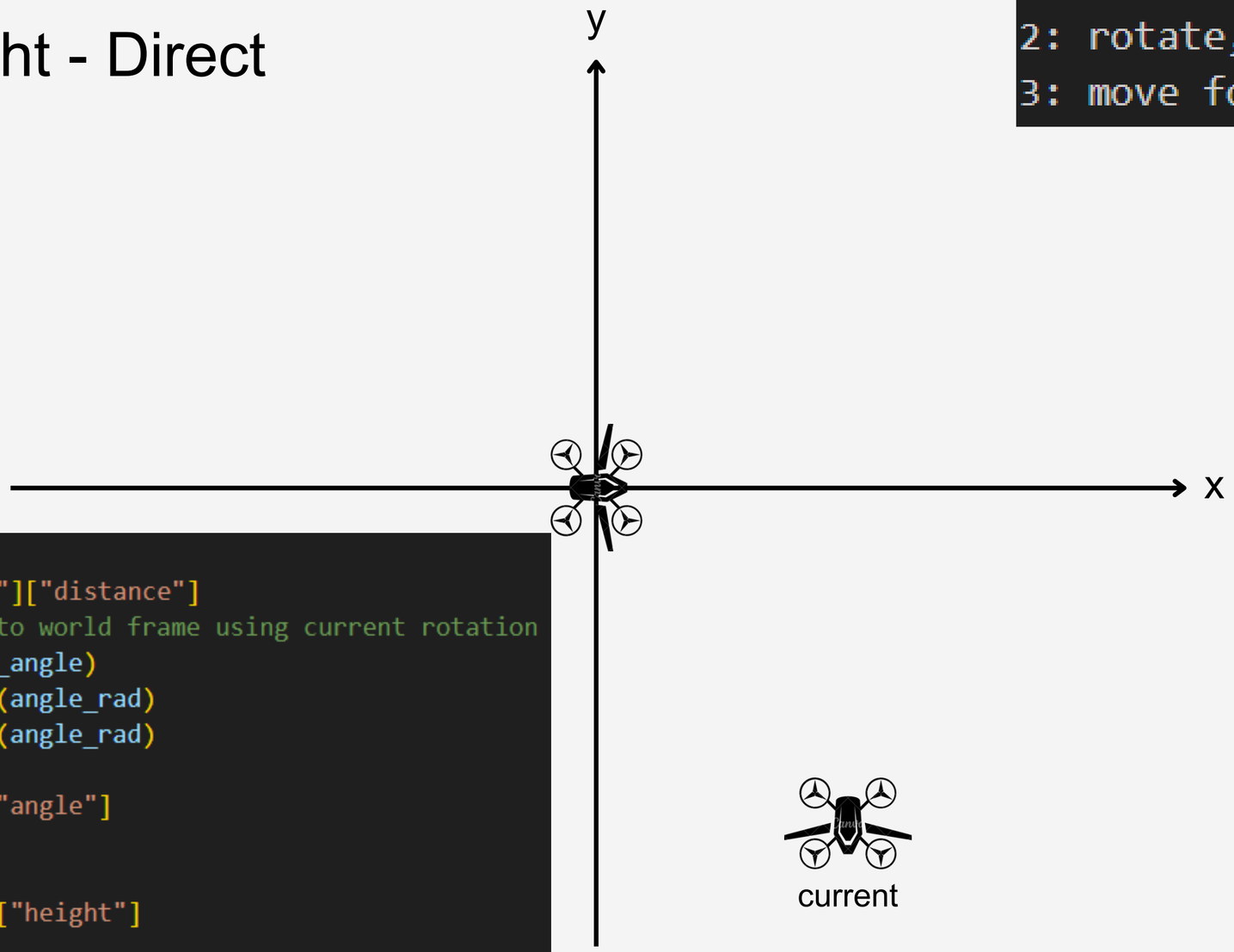


Return Flight - Direct

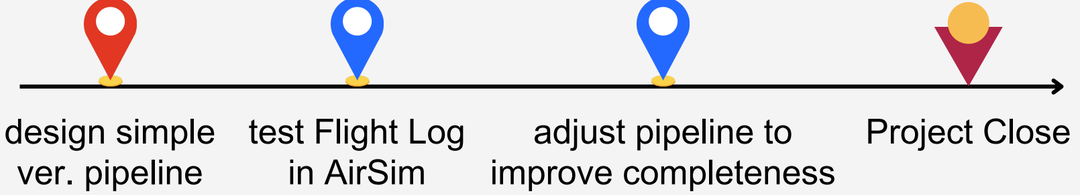
```
1: move forward, distance: 50
2: rotate, angle: -90
3: move forward, distance: 80
```

$total_x = 0,\ total_y = 0$

$total_{\text{angle}} = 0$




```python
if action == "move_forward":
    distance = action_dict["params"]["distance"]
    # Convert body frame movement to world frame using current rotation
    angle_rad = math.radians(total_angle)
    total_x += distance * math.cos(angle_rad)
    total_y += distance * math.sin(angle_rad)
elif action == "rotate":
    angle = action_dict["params"]["angle"]
    total_angle += angle
elif action == "move_vertical":
    height = action_dict["params"]["height"]
    total_vertical += height
```



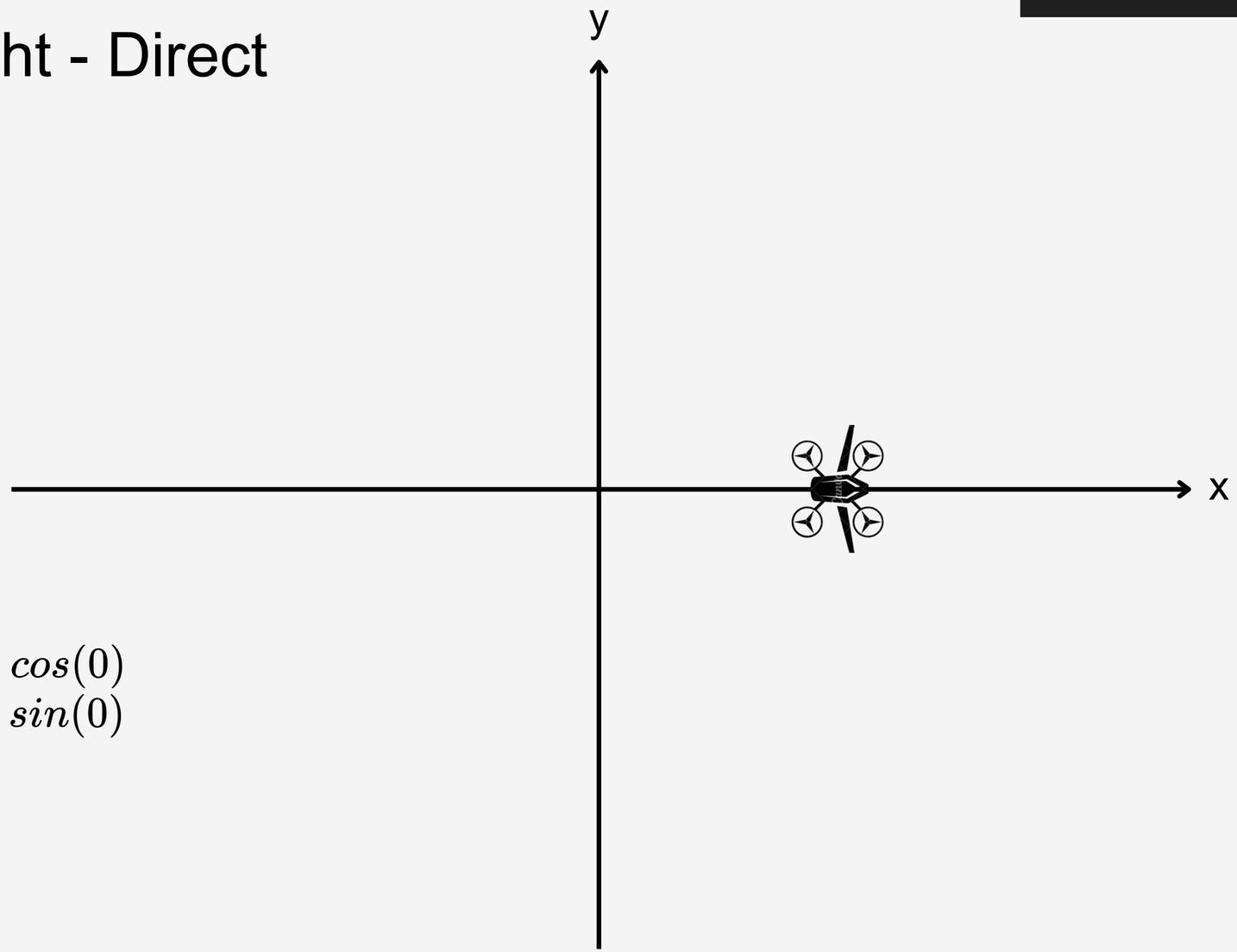
Methods - return flight

```
1: move forward, distance: 50
```

Return Flight - Direct

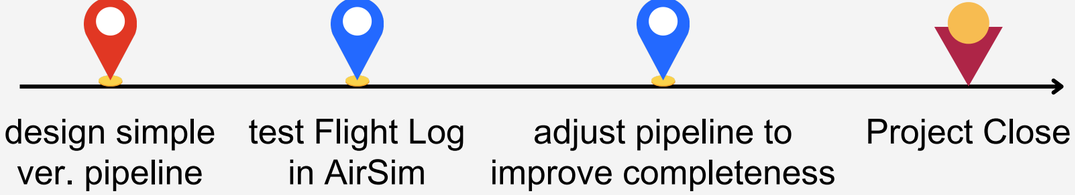
$total_x$ = 0, $total_y$ = 0

$total_{\text{angle}}$ = 0



What do we do?

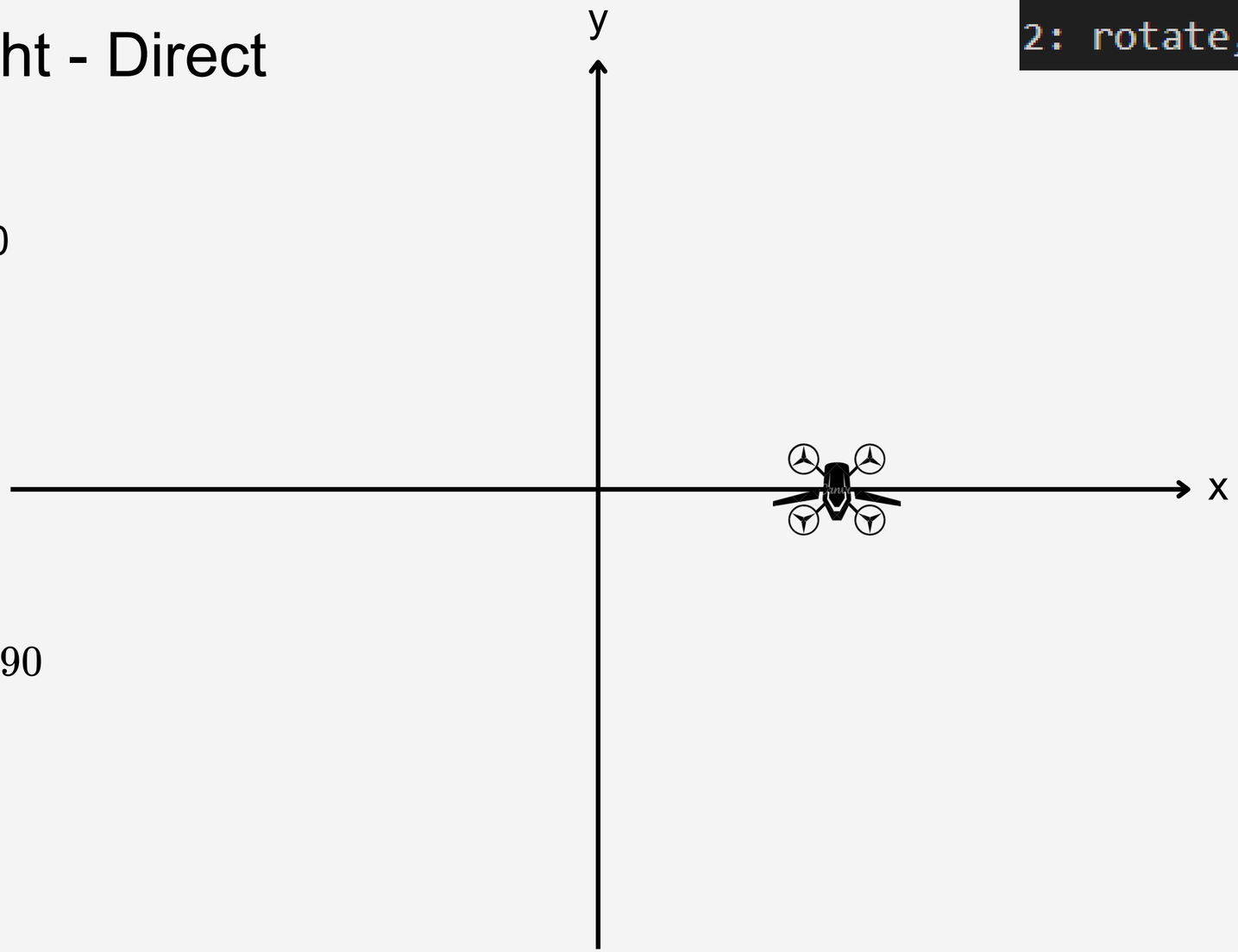
1. $total_x = 0 + 50 \cdot cos(0)$
2. $total_y = 0 + 50 \cdot sin(0)$



Methods - return flight

```
1: move forward, distance: 50
2: rotate, angle: -90
```

Return Flight - Direct



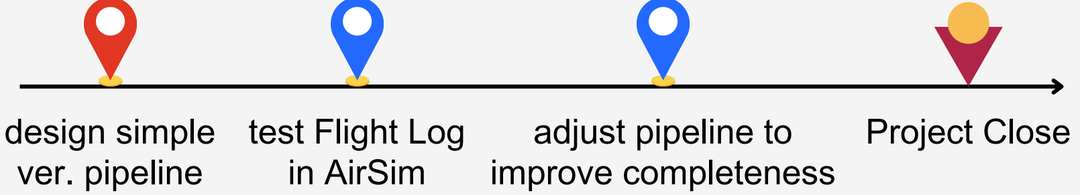
$total_x$ = 50, $total_y$ = 0

$total_{\text{angle}}$ = 0

What do we do?

1. $total_{\text{angle}} = 0 - 90$

Methods - return flight

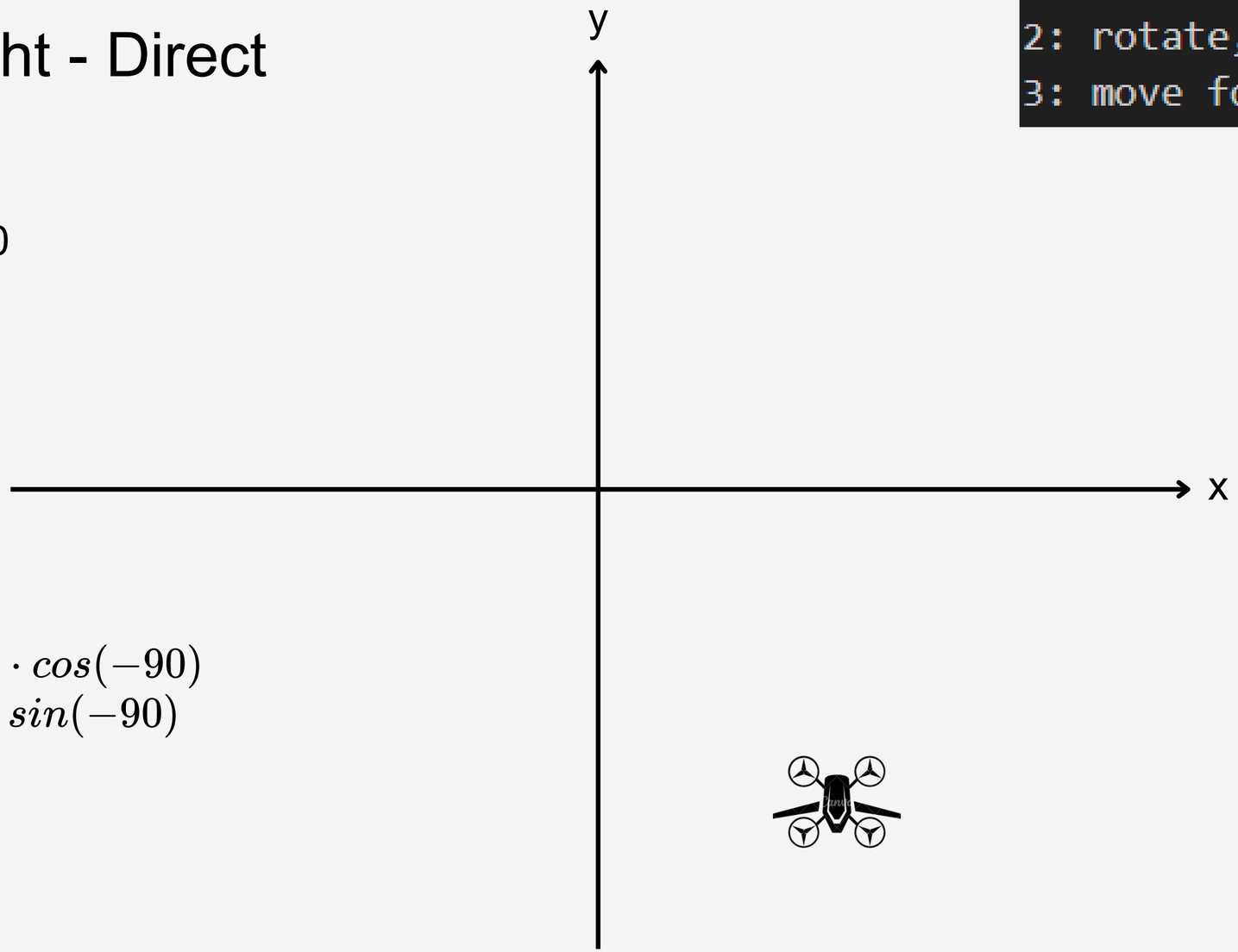



```
1: move forward, distance: 50
2: rotate, angle: -90
3: move forward, distance: 80
```

Return Flight - Direct

$total_x$ = 50, $total_y$ = 0

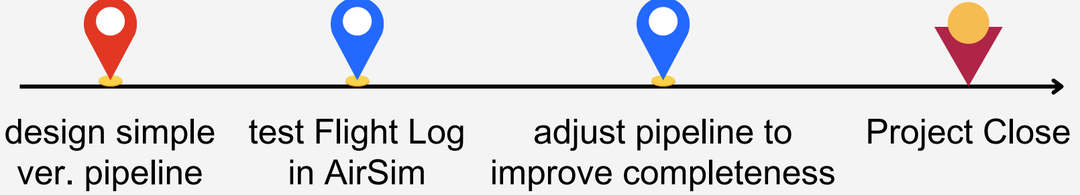
$total_{\text{angle}}$ = -90



What do we do?

1. $total_x = 50 + 80 \cdot cos(-90)$
2. $total_y = 0 + 80 \cdot sin(-90)$

Methods - return flight

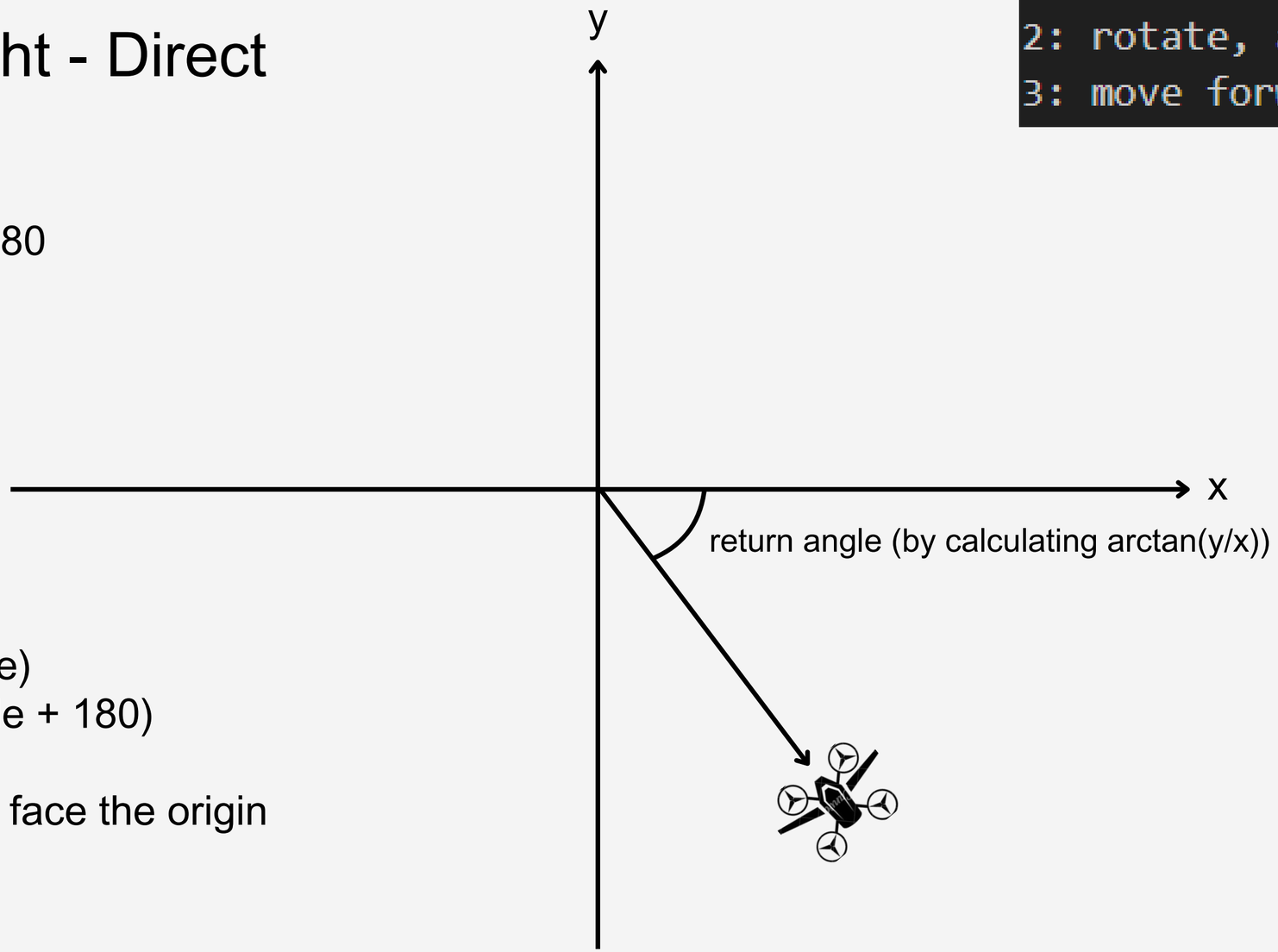


Return Flight - Direct

```
1: move forward, distance: 50
2: rotate, angle: -90
3: move forward, distance: 80
```

$total_x$ = 50, $total_y$ = -80

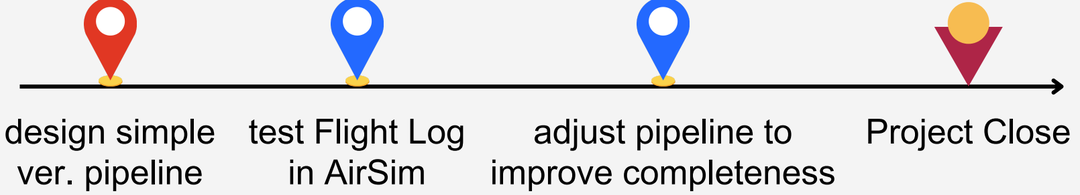
$total_{\text{angle}}$ = -90



What do we do?

1. rotate(-total_angle)
2. rotate(return angle + 180)

this makes the drone face the origin



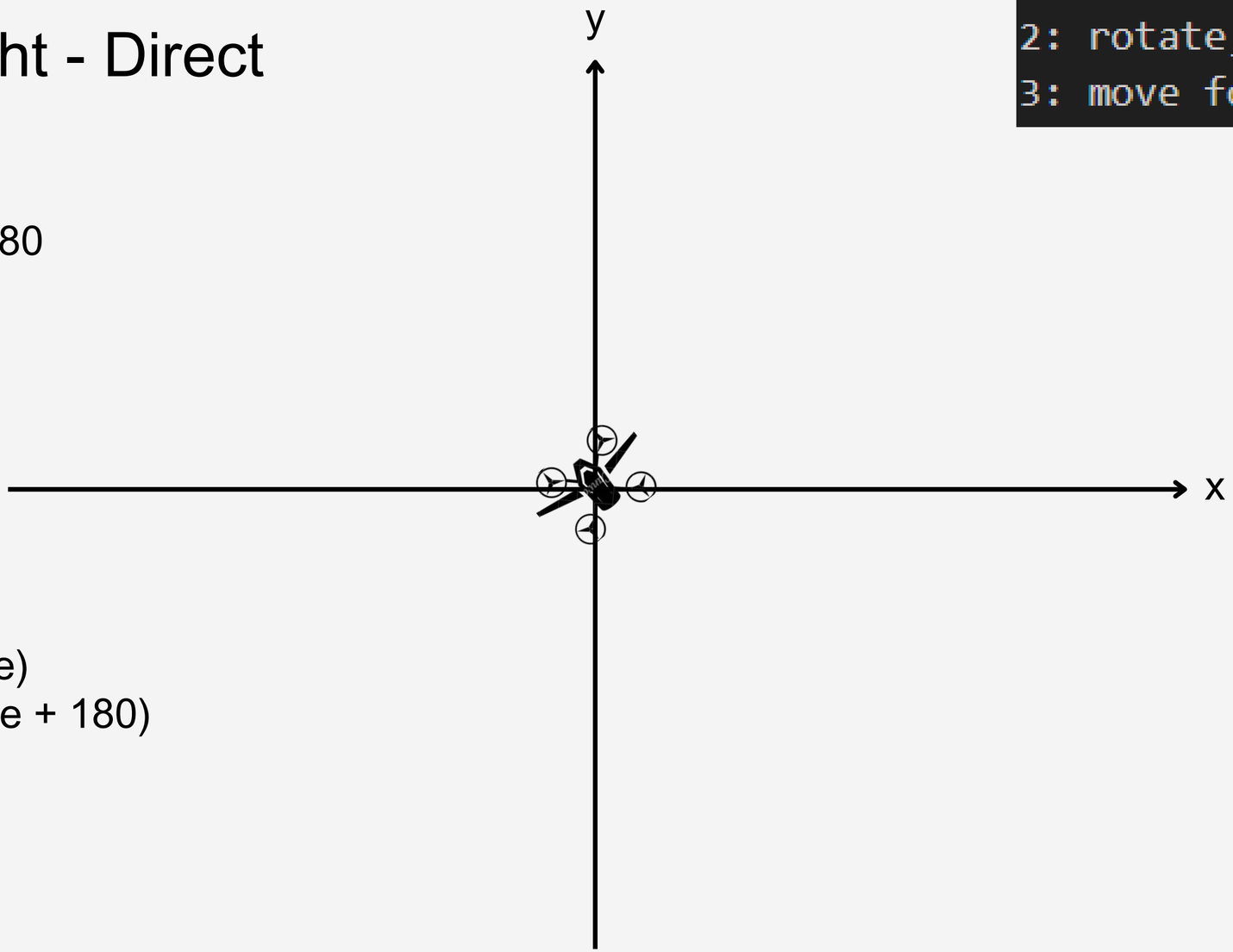
Methods - return flight

Return Flight - Direct

```
1: move forward, distance: 50
2: rotate, angle: -90
3: move forward, distance: 80
```

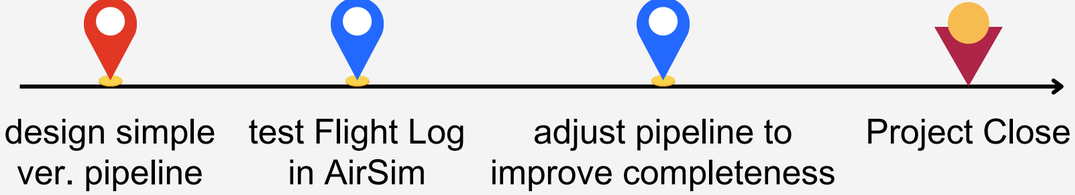
$total_x$ = 50, $total_y$ = -80

$total_{\text{angle}}$ = -90



What do we do?

1. rotate(-total_angle)
2. rotate(return angle + 180)
3. fly $\sqrt{x^2 + y^2}$



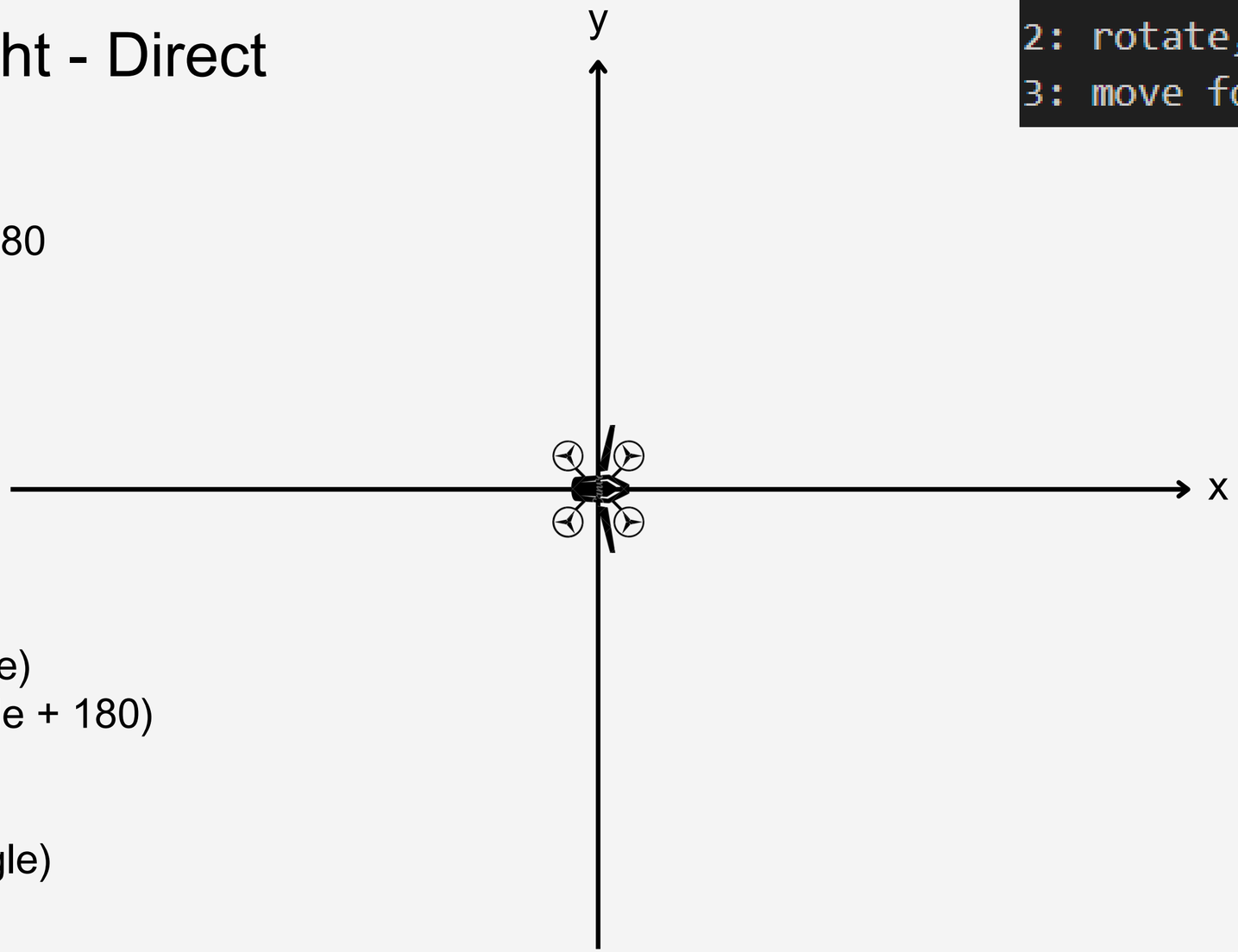
Methods - return flight

Return Flight - Direct

```
1: move forward, distance: 50
2: rotate, angle: -90
3: move forward, distance: 80
```

$total_x$ = 50, $total_y$ = -80

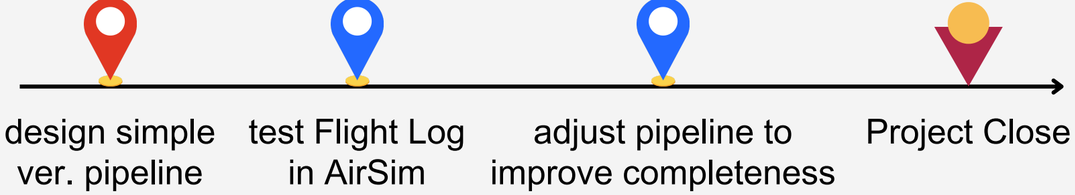
$total_{\text{angle}}$ = -90



What do we do?

1. rotate(-total_angle)
2. rotate(return angle + 180)
3. fly $\sqrt{x^2 + y^2}$
4. rotate(180)
5. rotate(-return angle)

Done!



Methods - return flight

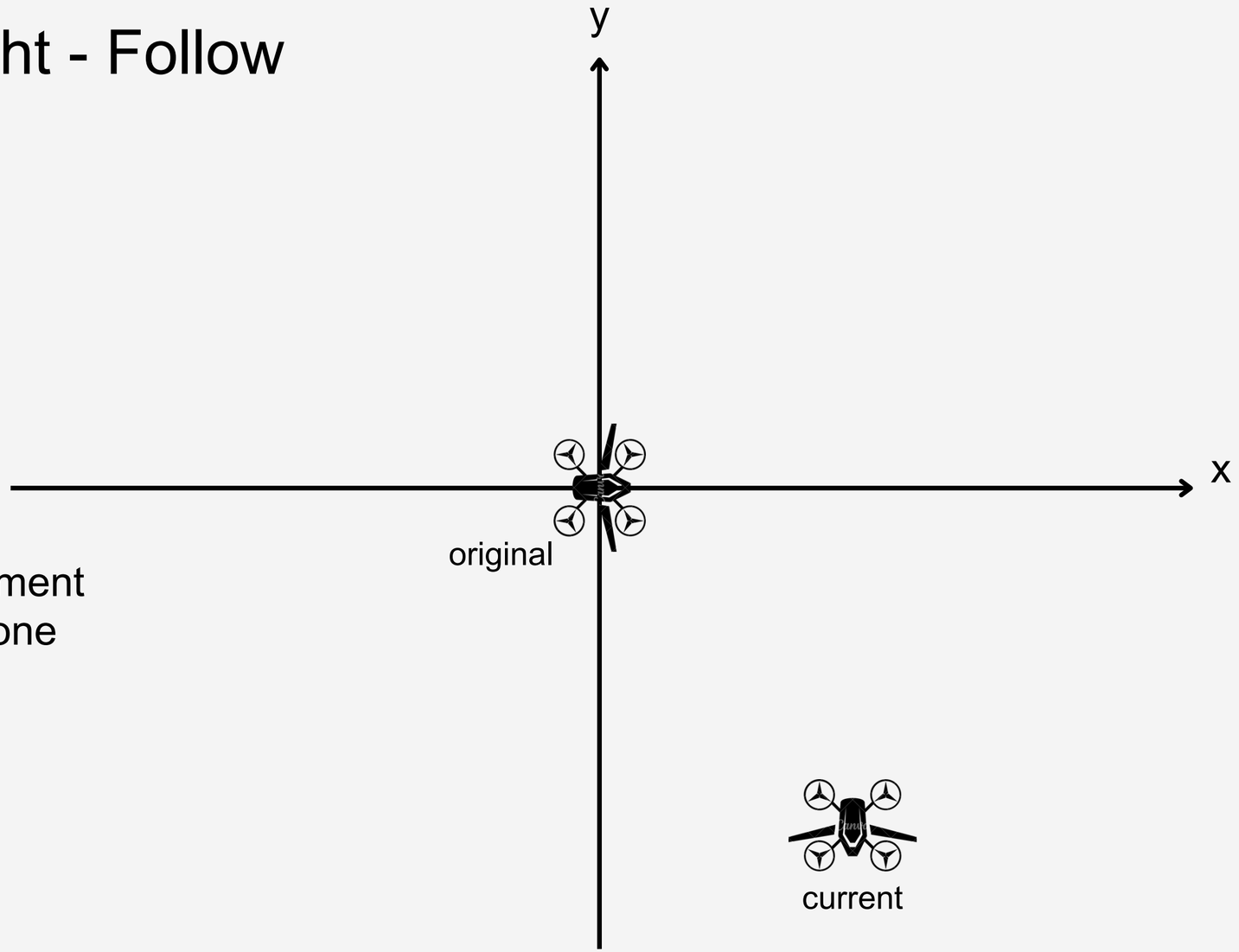
Return Flight - Follow

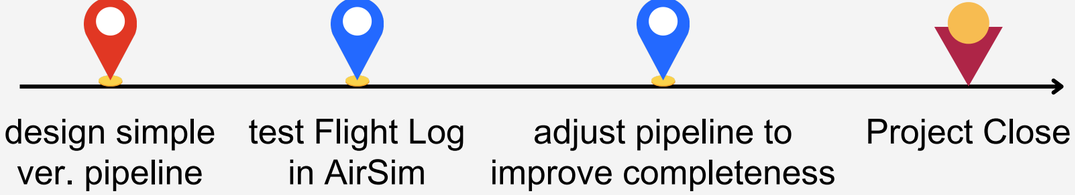
What we need:

1. Past Actions

Idea (Big picture):

1. simply take complement of the past actions one by one



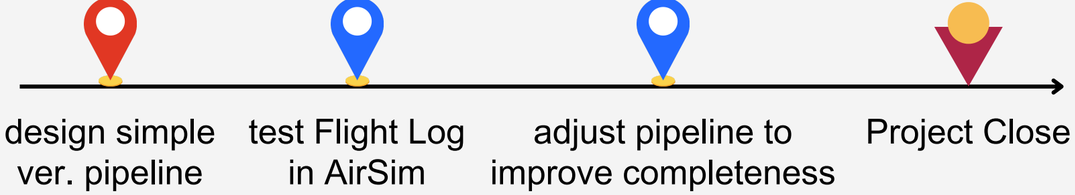


Methods - return flight

Return Flight - Follow

Fly forward <--> rotate 180 + Fly forward + rotate 180
rotate x <--> rotate -x
ascend x <--> ascend -x

Notice that we can pend rotation degrees until we see “Fly forward” to optimize it.
(Don’t instant rotate when seeing rotate(x), accumulate it)

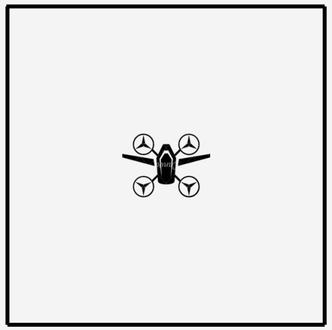


Methods - areal scan

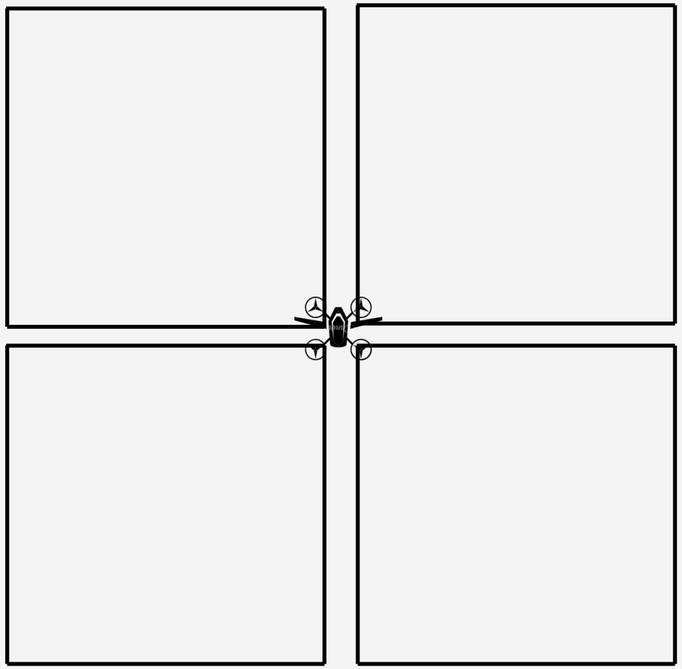
Areal Scan

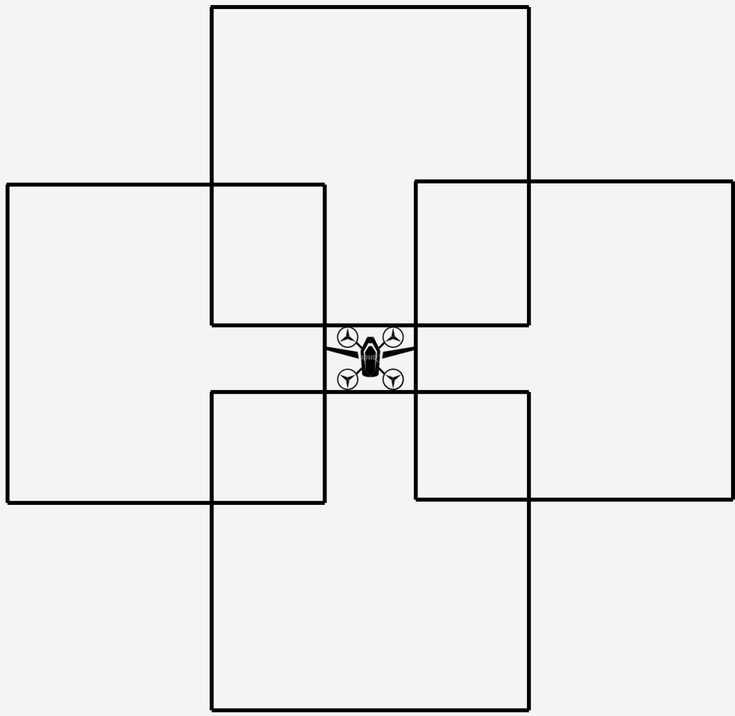
9 types of scanning method

What we need:
nothing

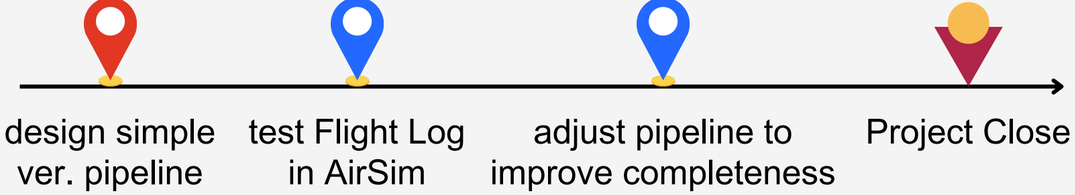


Spiral





Grid

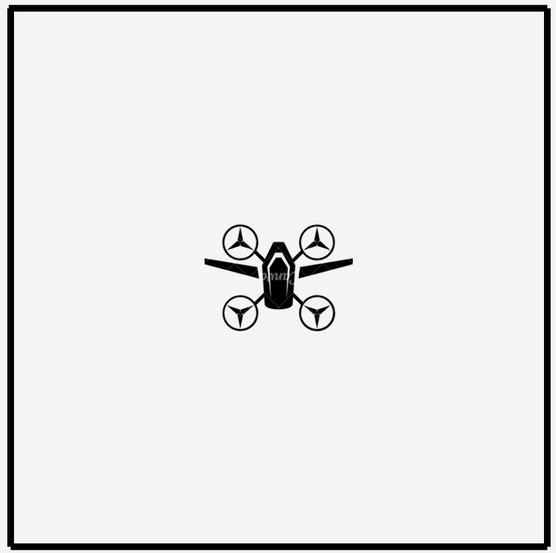


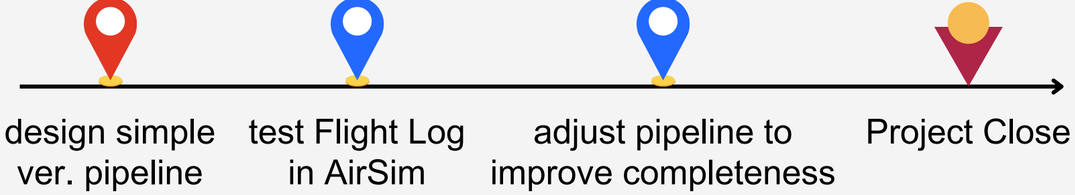
Methods - areal scan

Areal Scan - Spiral

Idea:

1. calculate the optimal spacing using camera FOV
2. num_rings = scan_radius / spacing
3. each ring = ⌜ + ⌟ (let's call this a 'L')

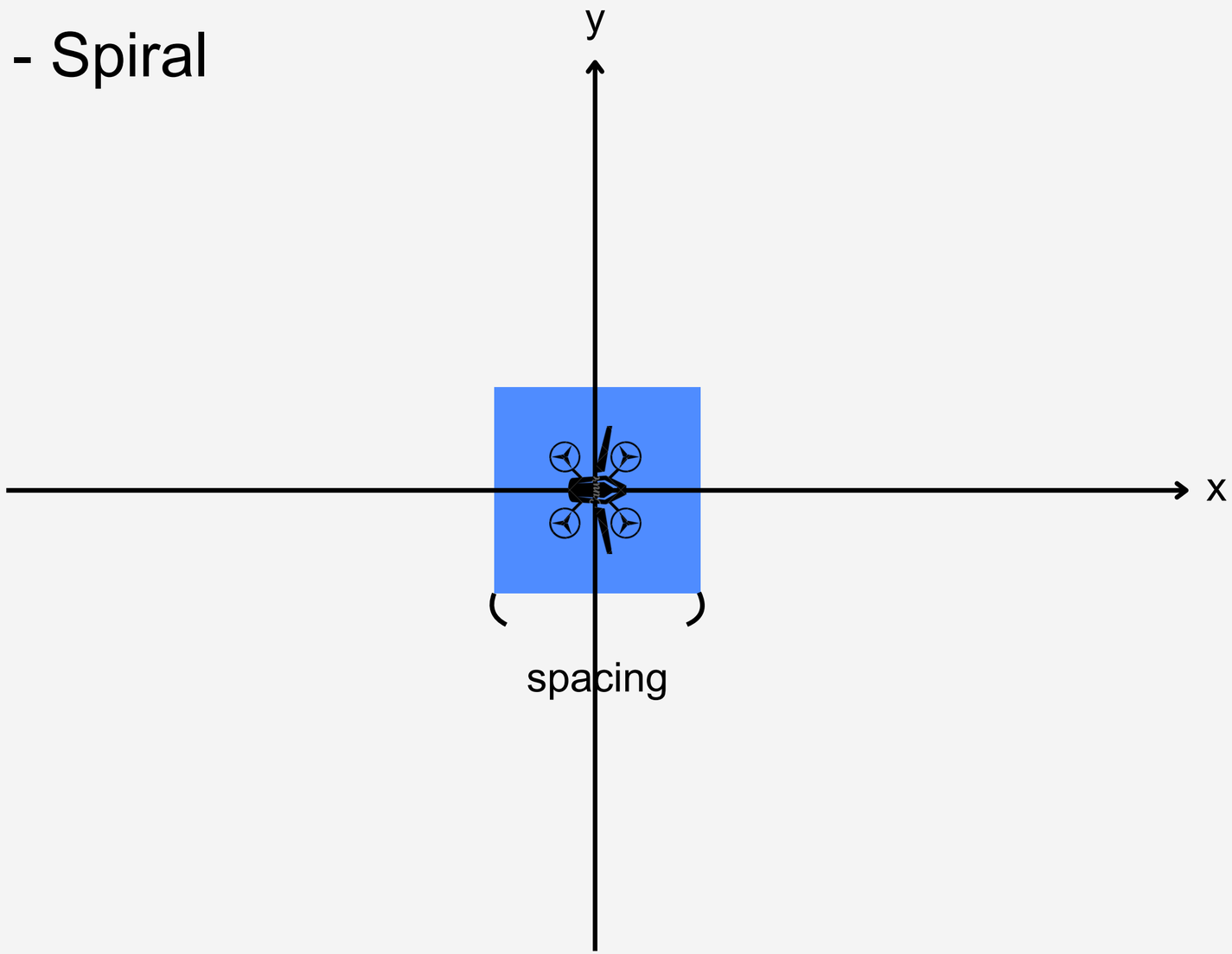


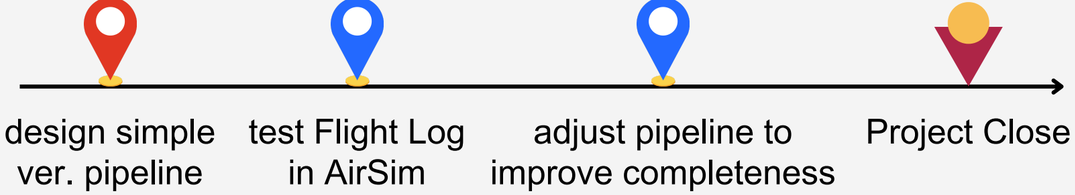


Methods - areal scan

Areal Scan - Spiral

L number 0

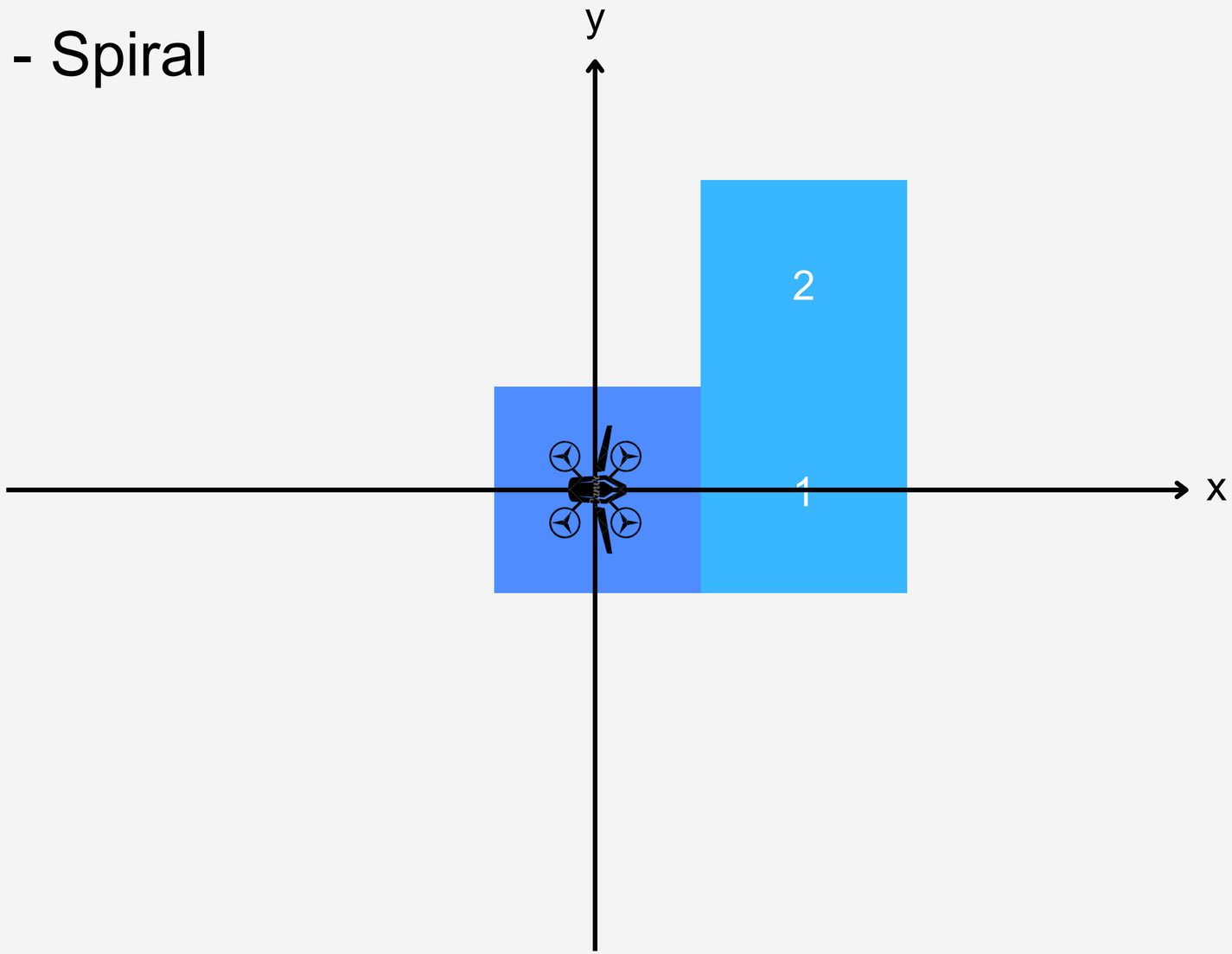


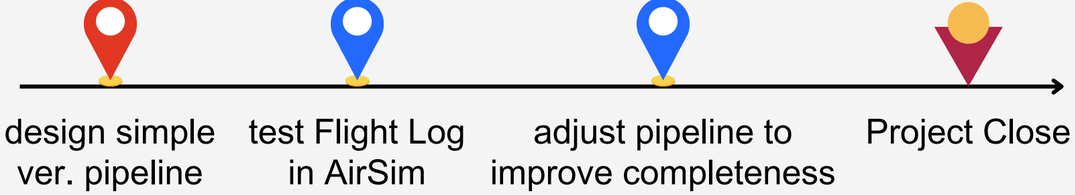


Methods - areal scan

Areal Scan - Spiral

L number 1

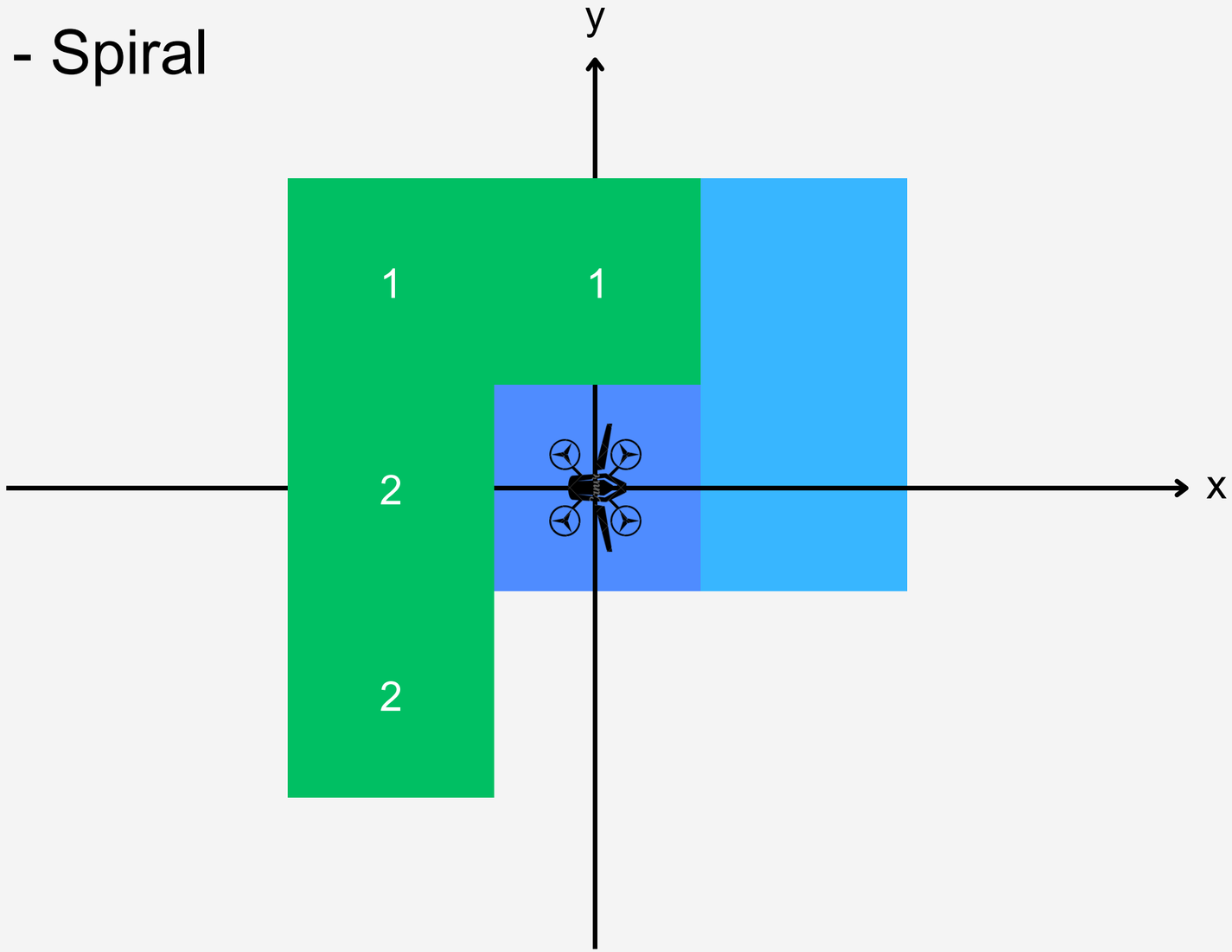


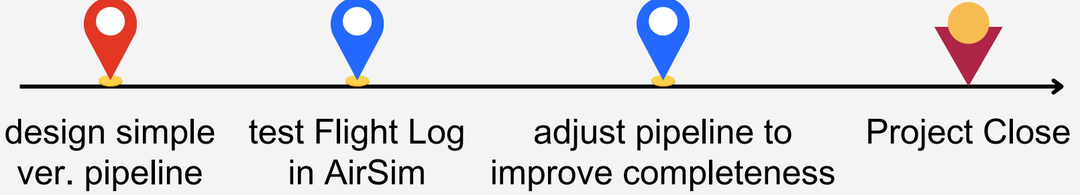


Methods - areal scan

Areal Scan - Spiral

L number 2

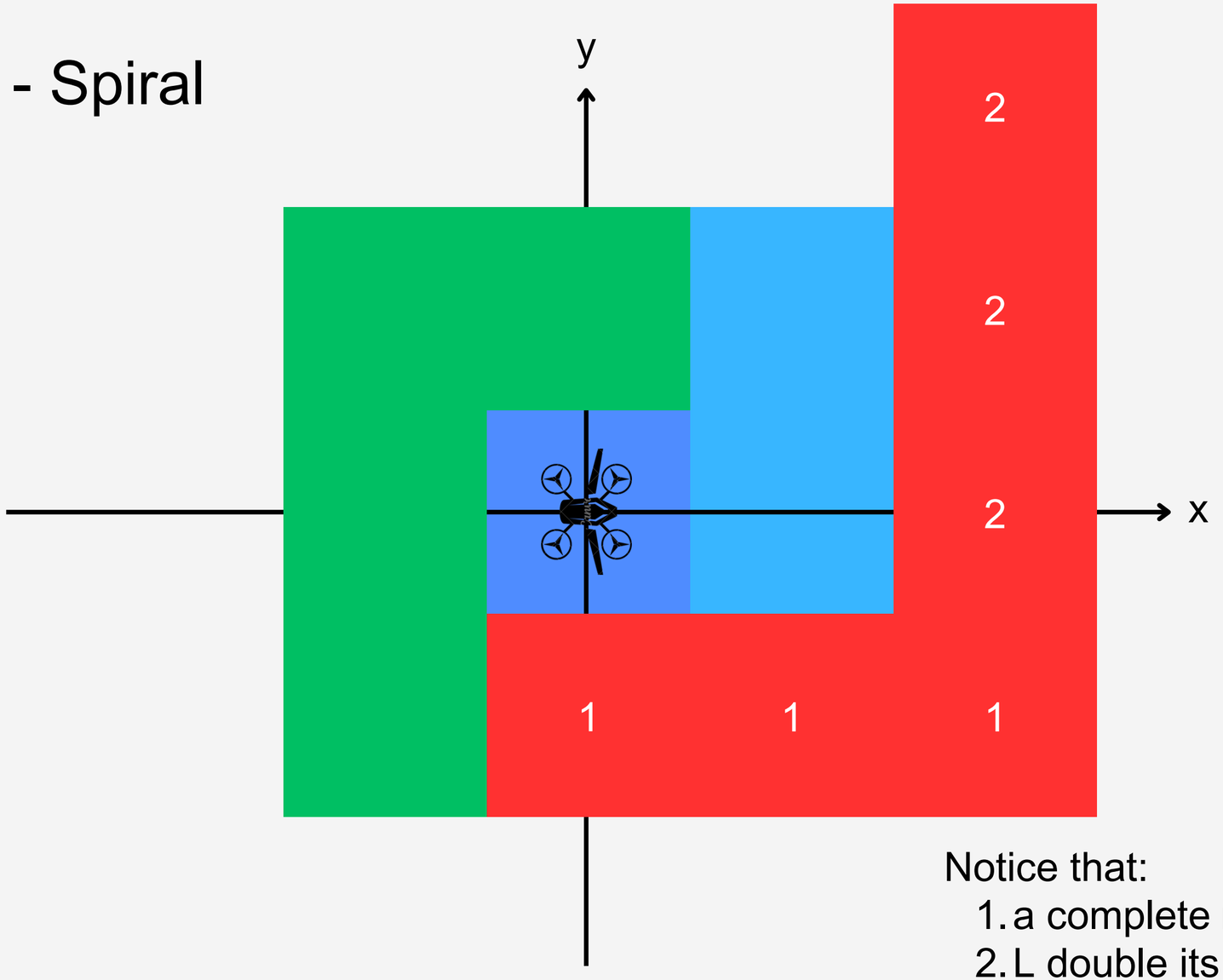




Methods - areal scan

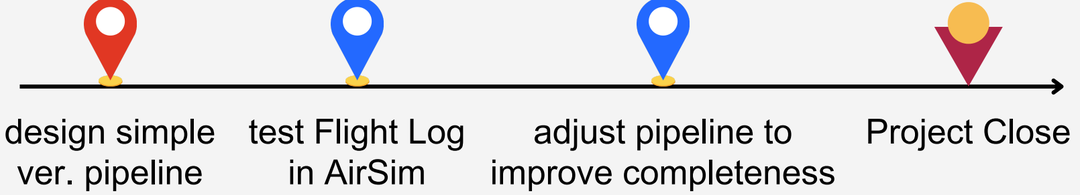
Areal Scan - Spiral

L number 3



Notice that:

1. a complete ring always end on odd L
2. L double its length

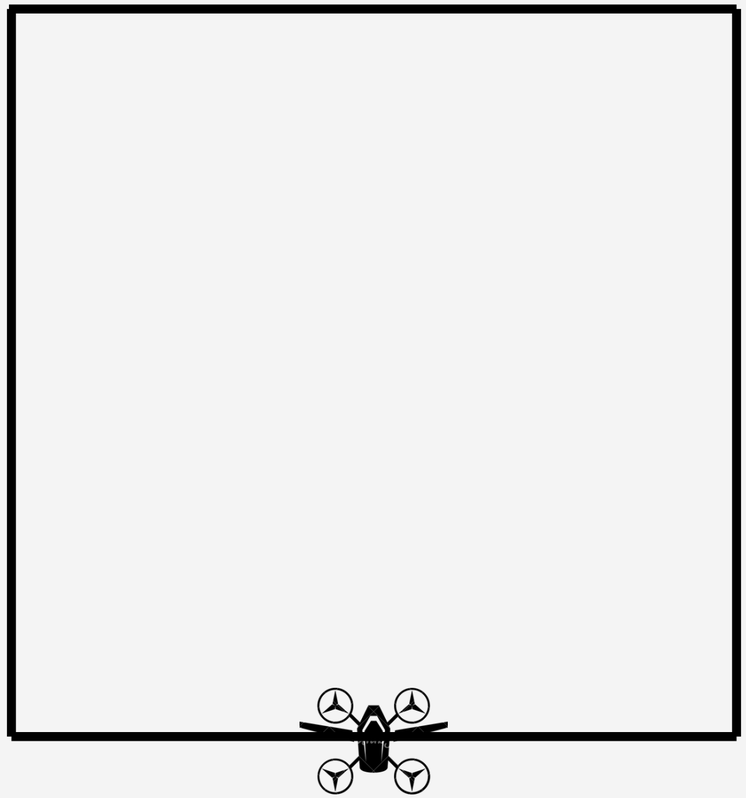


Methods - areal scan

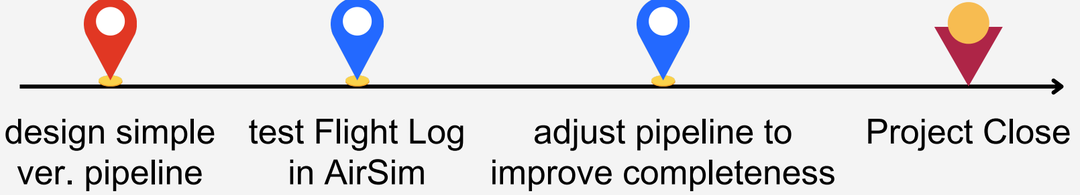
Areal Scan - Grid

Idea:

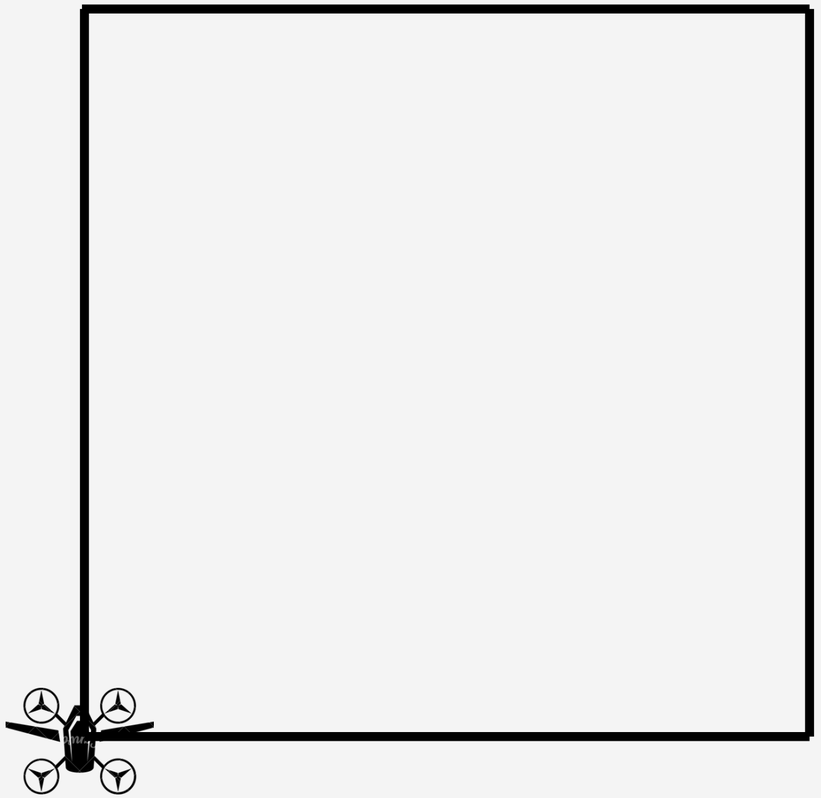
1. calculate the optimal spacing using camera FOV
2. decide number of legs
3. move the drone to the bottom left corner
4. start scanning

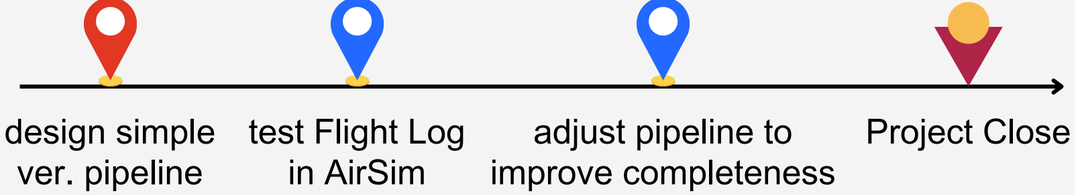


Methods - areal scan



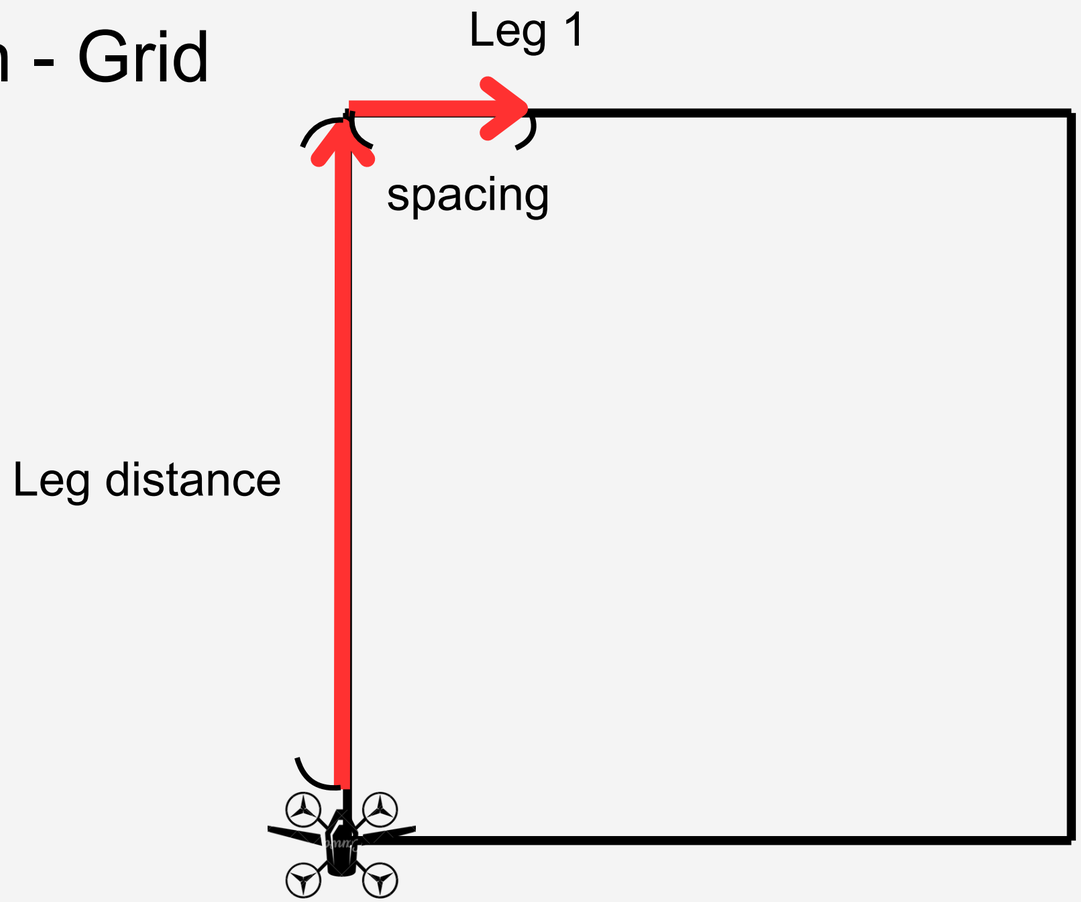
Areal Scan - Grid

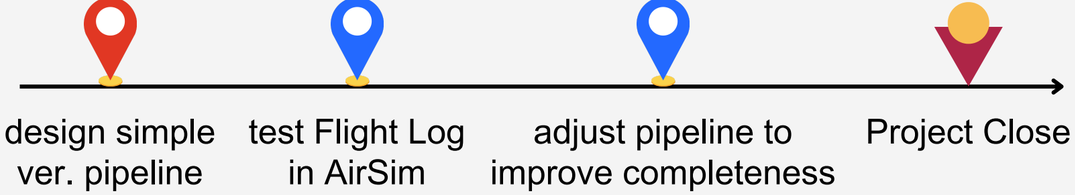




Methods - areal scan

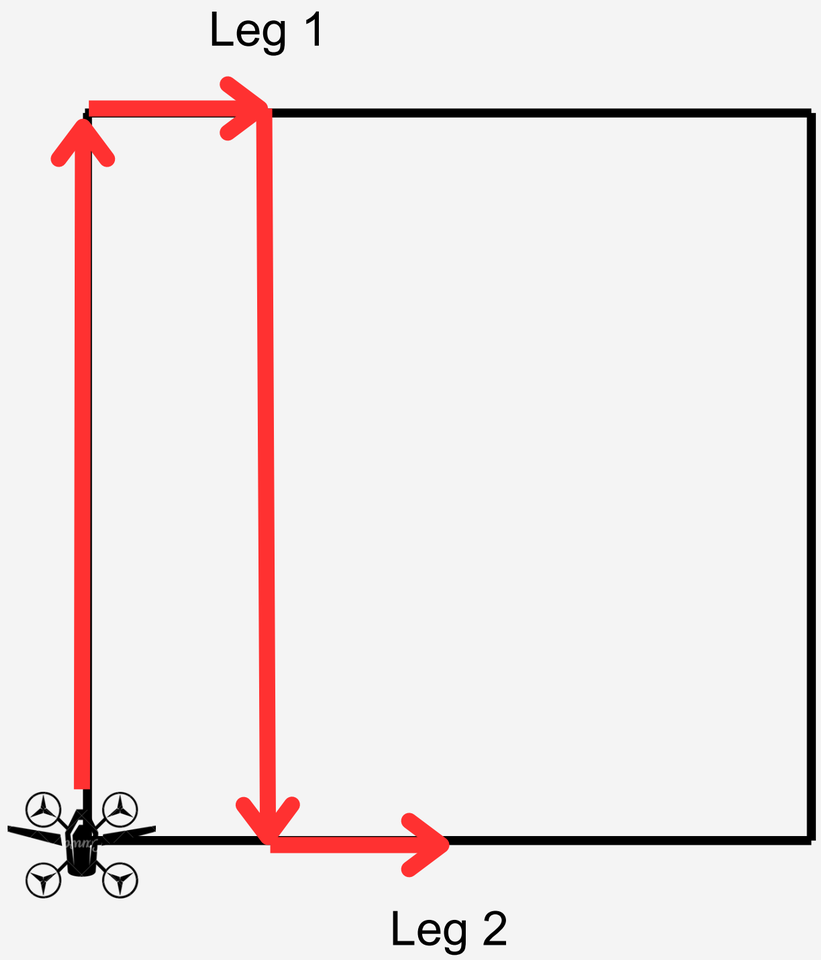
Areal Scan - Grid

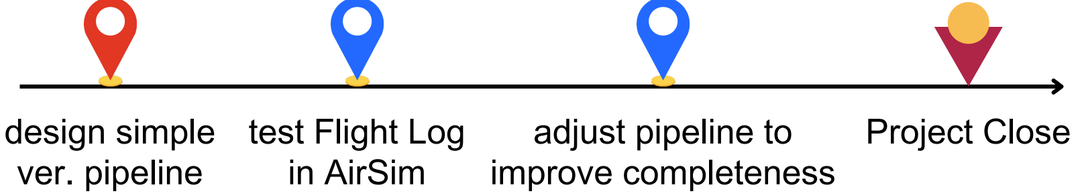




Methods - areal scan

Areal Scan - Grid



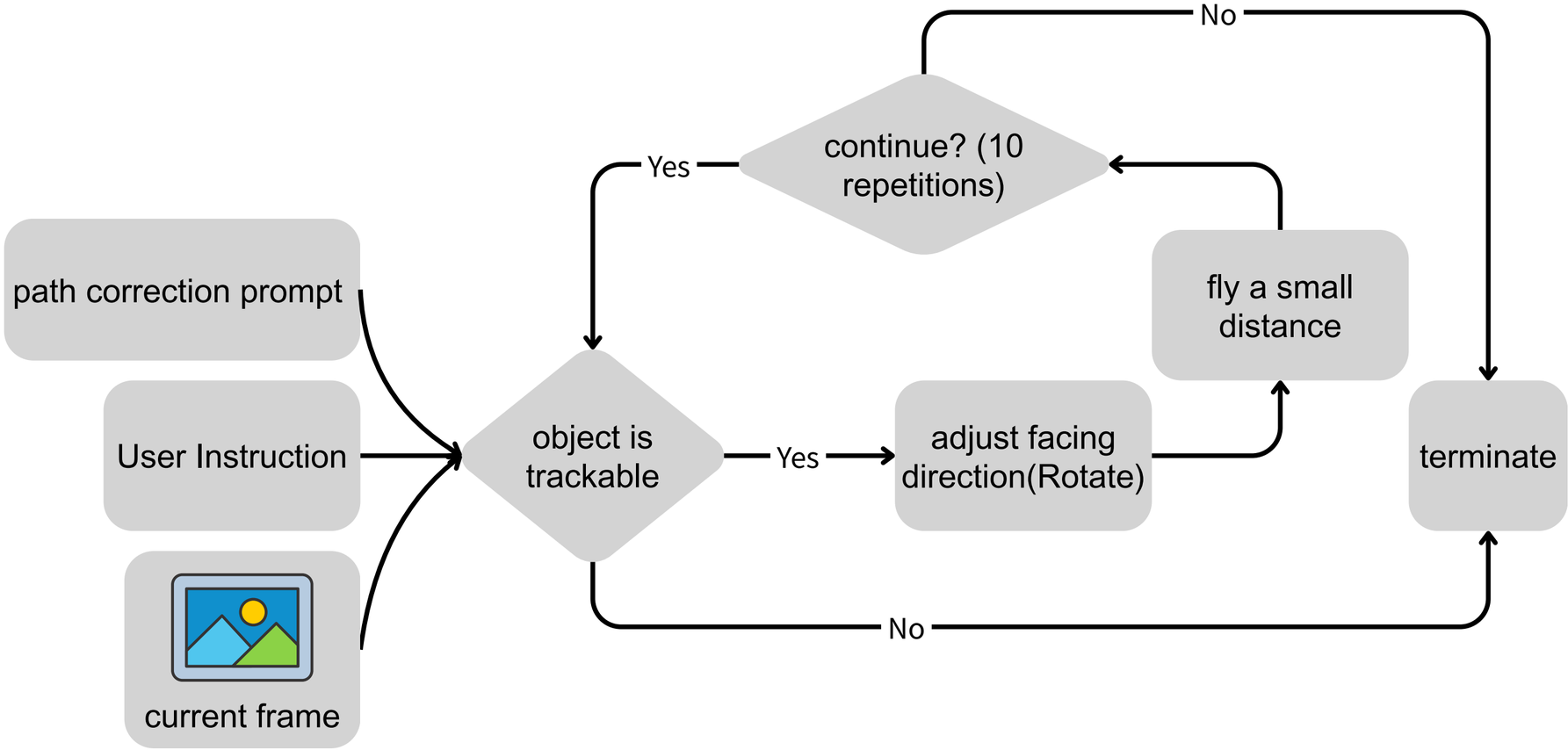


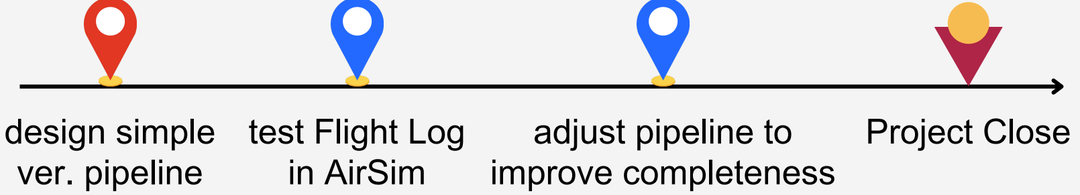
Methods - follow path

Follow Path (Static Object)

What we need:

1. Current Frame
2. Path Correction prompt

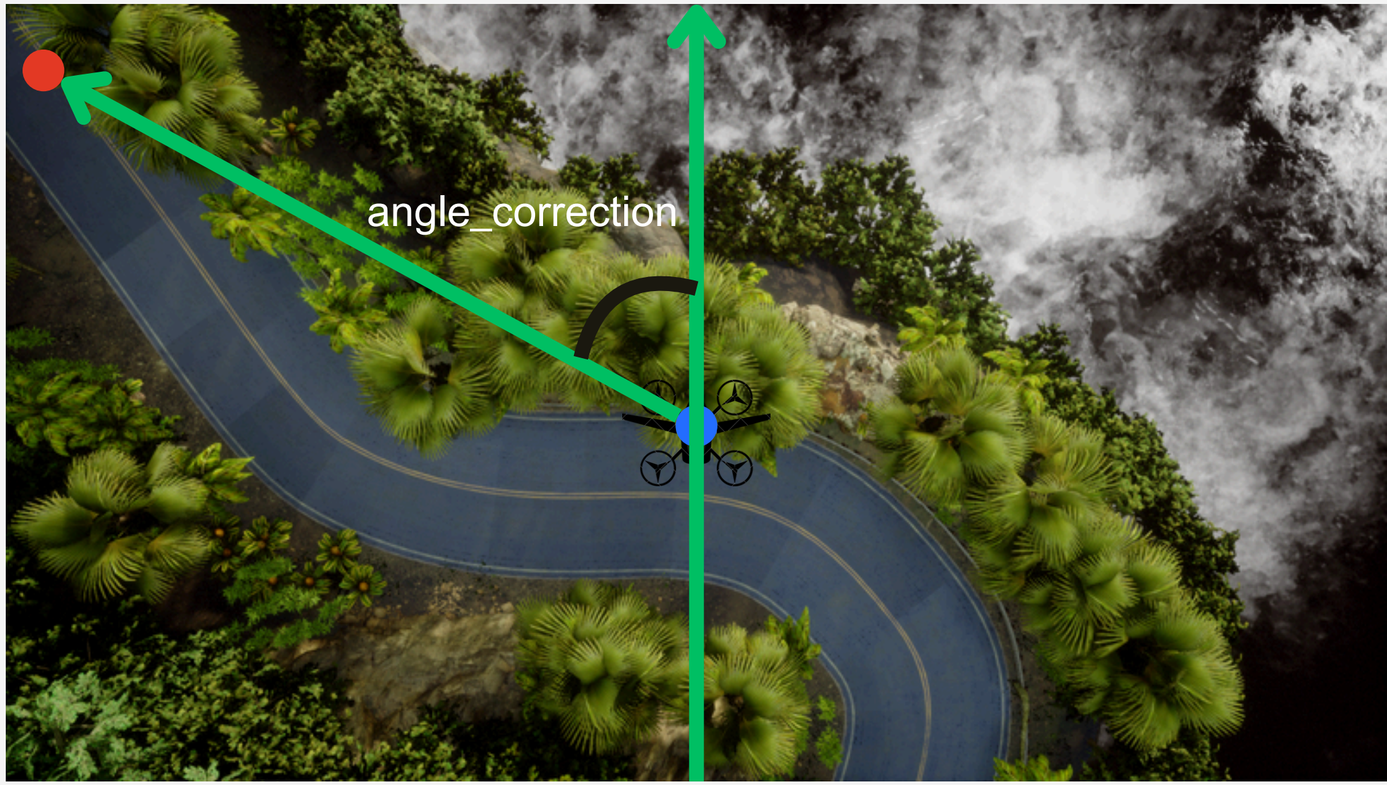




Methods - follow path

Follow Path (Static Object)

The Path Correction prompt tells the model:
1. Check if that object exists in the frame.
2. If yes, find its vanishing point (topmost direction it continues to) and compute the angle correction from the drone at the frame center.
3. If not found, explain why.



Result

The following result will be demonstrated in pictures. For a better experience, visit the GitHub page to check out the videos.

Result - Flight Log Generation

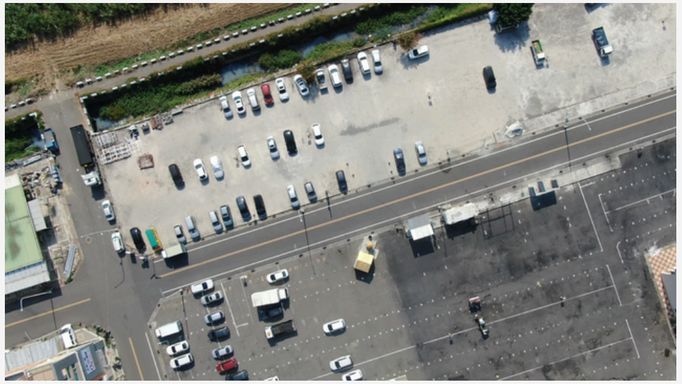
In this section, let’s focus on Flight Logs generated from real-world drone footage. Including our own and those downloaded from the internet:

- 嘉義(Jiayi City, Taiwan), video link
- Ice mountain, video link
- Following a police car, video link
- Industrial zone, video link
- Protest street, video link

The following pages will only show the result of “Jiayi City” and “Following a police car”.

Result - Flight Log Generation

嘉義(Jiayi City, Taiwan)



Result - Flight Log

嘉義(Jiayi City, Taiwan)

DRONE FLIGHT LOG

FLIGHT IDENTIFICATION

Date: 2025-08-19
Start Time: 10:00
End Time: 10:05
Total Duration: 5 min 1 sec

FLIGHT PURPOSE & OPERATIONS

Purpose of Flight: City Safety Monitoring
Type of Operation: Aerial Surveillance

LOCATION & ENVIRONMENT

Location Name/Description: Taiwan Jiayi - Mixed urban/suburban and agricultural area including sports complex, parking lots, buildings, roads, and agricultural fields
GPS Coordinates (Takeoff): Empty sporting ground
GPS Coordinates (Landing): Same as takeoff place
Weather Conditions:

- Wind Speed: Low
- Wind Direction: Not mentioned
- Visibility: Excellent - clear conditions with bright sunlight
- Temperature: Sunny, likely warm
- Cloud Cover: Clear skies
- Precipitation: None

FLIGHT PARAMETERS

Maximum Altitude: Not visible
Maximum Distance: Not visible
Flight Pattern: ☑ Free Flight
Key Waypoints/Locations: Sports courts, baseball field, building complex with solar panels, agricultural fields with crops, parking areas, main roads with intersection, abandoned buildings
Flight Path Summary: Initial vertical ascent over sports court, followed by horizontal movement through building complex with red roofs and solar panels, over agricultural fields, parking lots, abandoned buildings, and ended at sports facilities, maintaining primarily a top-down perspective with good altitude changes and camera movements

CAMERA & RECORDING SETTINGS

Video Resolution: 1920x1080
Frame Rate: 29.97 fps
Recording Format: mp4

SAFETY CONSIDERATIONS

Obstacles Present: ☑ Trees ☑ Buildings ☑ Power Lines ☑ Other: Fencing around sports facilities
People in Area: ☑ Pilot Only ☑ Small Group
Emergency Landing Sites: Sports courts, baseball field, large parking lots, open agricultural fields

INCIDENTS & OBSERVATIONS

Any Issues Encountered: Initial focus problems with blurry footage that quickly cleared
Wildlife Interactions: None
Signal Loss Events: None
Weather Changes: None
Equipment Malfunctions: Initial blurry footage in first frame (quickly resolved)

NOTES & LESSONS LEARNED

Flight Performance: Stable, smooth flight with good altitude changes and camera movements
Footage Quality: Clear and high resolution after initial focus issues, good detail throughout most of flight
Areas for Improvement: Initial camera focus calibration
Future Considerations: Monitor for people in sports areas, maintain safe distance from buildings with solar installations. For city safety monitoring purposes, establish regular flight patterns over key infrastructure areas including sports facilities, institutional buildings, and transportation corridors. Consider implementing automated waypoint missions to ensure consistent coverage of critical safety zones. Monitor parking areas and road intersections for traffic flow analysis and emergency response planning.

quite accurate!

Result - Flight Log Generation

Following a police car

Result - Flight Log (

Following a police car

DRONE FLIGHT LOG

FLIGHT IDENTIFICATION

Date: 2025-08-20
Start Time: 10:00
End Time: 10:05
Total Duration: 4 min 41 sec

FLIGHT PURPOSE & OPERATIONS

Purpose of Flight: observe and document police activity during a road check
Type of Operation: Aerial Surveillance Flight

LOCATION & ENVIRONMENT

Location Name/Description: America, country road. The road is a narrow, paved two-lane road with white markings, winding through a landscape of rolling, eroded hills covered in dense, low-lying green shrubbery and trees. There are power lines and poles running alongside parts of the road. Some cultivated fields are visible further along the path.
GPS Coordinates (Takeoff): police station
GPS Coordinates (Landing): same as takeoff place
Weather Conditions:

- Wind Speed: Low (vegetation shows minimal movement)
- Wind Direction: Not mentioned
- Visibility: Clear conditions (bright lighting throughout)
- Temperature: Not mentioned
- Cloud Cover: Overcast or hazy sky, no distinct clouds visible / Clear day with minimal cloud cover
- Precipitation: None

FLIGHT PARAMETERS

Maximum Altitude: Approximately 50-100 meters (estimated, based on perspective relative to trees and hills)
Maximum Distance: Not mentioned
Flight Pattern: ☑ Free Flight
Key Waypoints/Locations: Initial scene with a tractor, police car, and people; rural road through lush landscape, bridge/culvert structure/elevated sections, cultivated fields, areas with cattle/herd of buffalo on the road
Flight Path Summary: The drone begins at a lower altitude, observing a stationary tractor and police car with people gathered around. It then ascends slightly and follows the police car as it drives along a winding rural road through dense greenery, elevated sections, and agricultural areas. The drone maintains a relatively consistent altitude and follows the road through the vegetated, hilly terrain, encountering a motorcycle and later livestock.

CAMERA & RECORDING SETTINGS

Video Resolution: 3840x2160
Frame Rate: 29.97 fps / 29.97002997002997 fps
Recording Format: mp4

SAFETY CONSIDERATIONS

Obstacles Present: ☑ Trees ☑ Power Lines ☑ Other: Rolling, eroded hills/ravines, steep embankments, utility poles, other vehicles (tractor, motorcycle), and livestock
People in Area: ☑ Small Group (police officers, person standing by vehicle, initially near the tractor/car, then individuals on motorcycle and walking)
Emergency Landing Sites: The paved road itself, open cultivated fields visible throughout flight path, or relatively flat, open patches of ground adjacent to the road

INCIDENTS & OBSERVATIONS

Any Issues Encountered: None mentioned/None visible
Wildlife Interactions: A herd of cattle/buffalo was encountered on the road, scattered across pavement interacting with police vehicle, causing the police car to slow down
Signal Loss Events: None mentioned/None visible
Weather Changes: Lighting transitions from early morning/late afternoon golden hour to dusk with purple hues / No significant weather changes observed during the flight
Equipment Malfunctions: None mentioned/None visible

NOTES & LESSONS LEARNED

Flight Performance: The flight appears stable and smooth, with good control over altitude and movement. Steady tracking of police vehicle throughout rural terrain
Footage Quality: High quality 4K footage with consistent framing and smooth movement. The footage is clear, well-focused, and well-exposed throughout the duration
Areas for Improvement: Not mentioned
Future Considerations: For police activity documentation flights, establish multiple predetermined observation altitudes to capture both wide area context and detailed vehicle interactions. Consider coordination protocols when wildlife (cattle) may interfere with traffic stops. Plan for extended flight times during rural patrol documentation as distances between incidents may be greater than urban operations.

did mention cultivated field, but didn't see the lawnmower

Result - Autonomous Flight

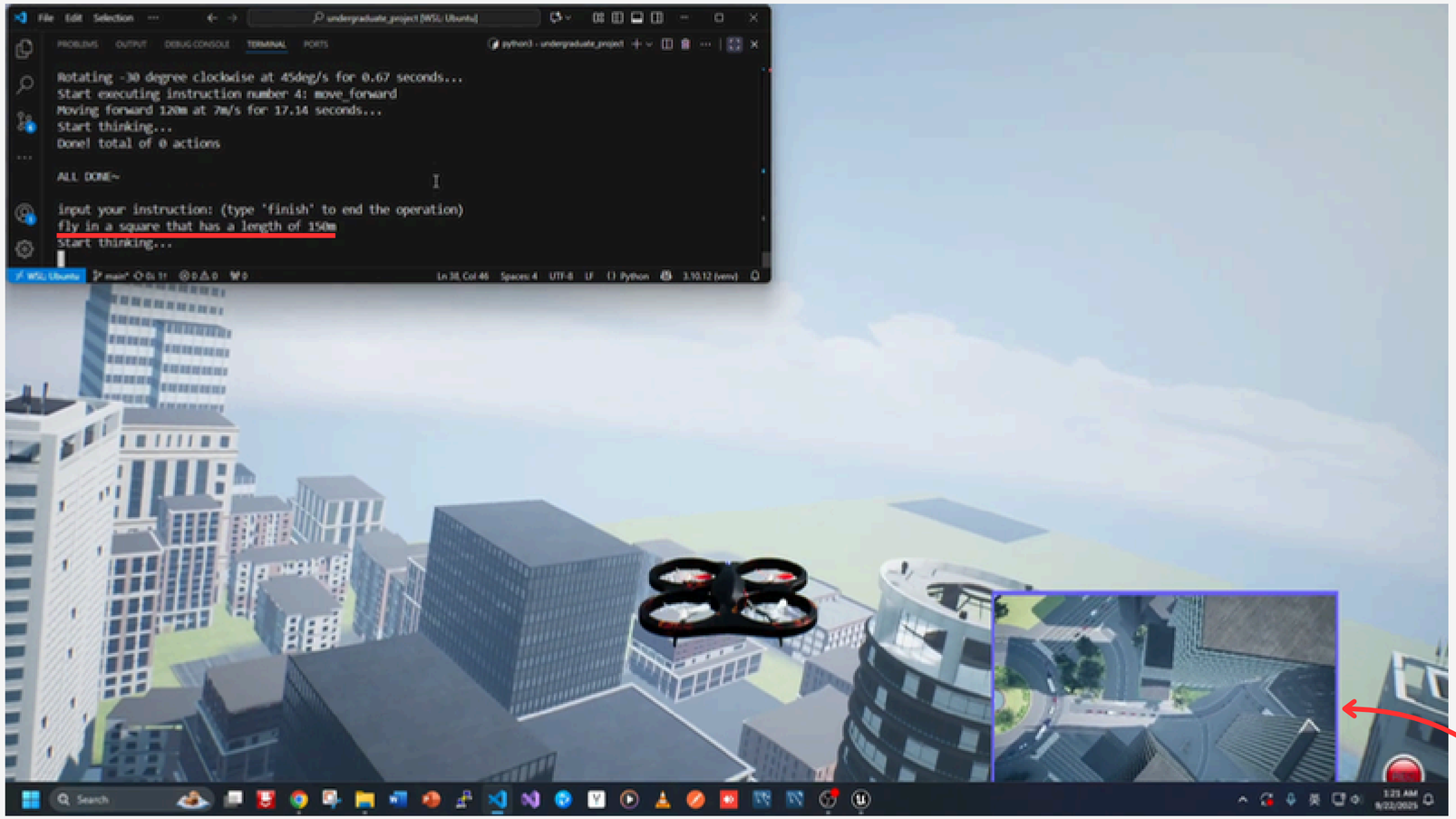
In this section, let’s focus on the auto pilot implementation in AirSim:

- case 1: Simulate Private recreational area (containing newest predefined abstract missions)
  - demo video on YouTube
  - actual drone footage
  - Flight Log report
- case 2: Simulate City
  - demo video on YouTube
  - actual drone footage
  - Flight Log report
- case 3: 張家界 (ZhangJiajie)
  - demo video on YouTube
  - actual drone footage
  - Flight Log report

The following pages will only show one example for each level of instructions (Navigation, Predefined Mission, Vision Task) + one Flight Log generated from AirSim drone footage.

Result - Autonomous Flight

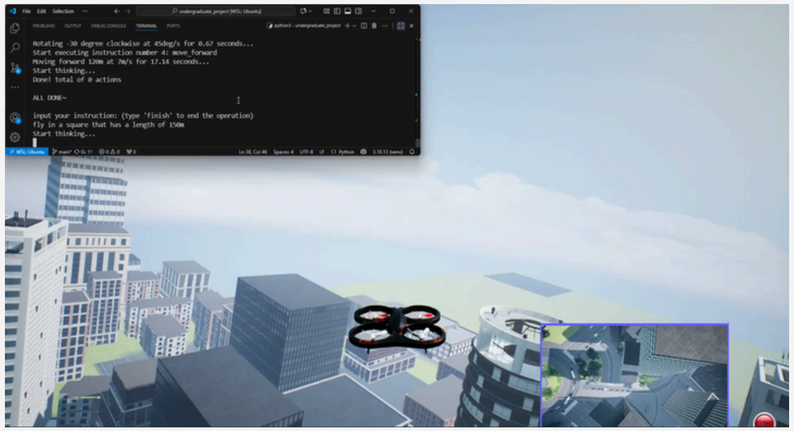
Navigation: “fly in a square that has a length of 150m”

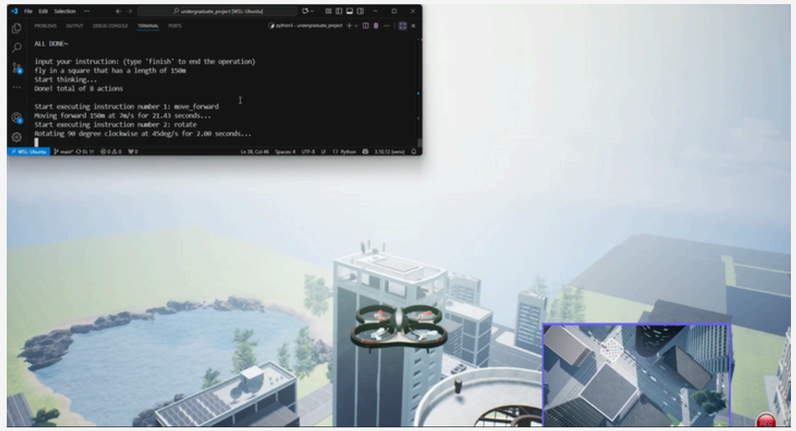


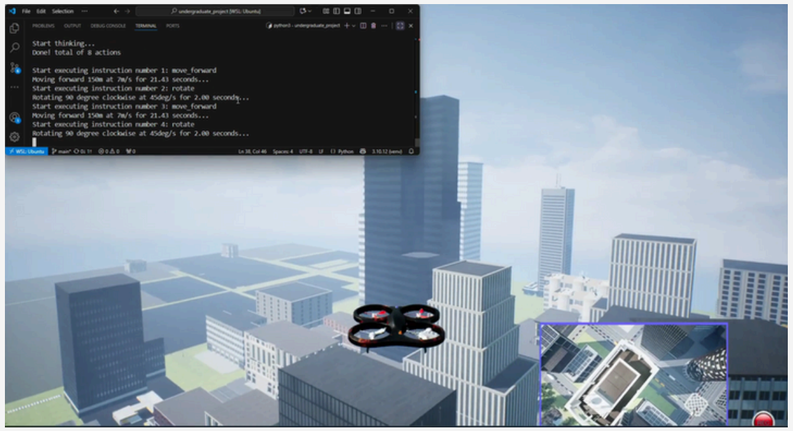
Focus on this, it’s what the drone actually sees

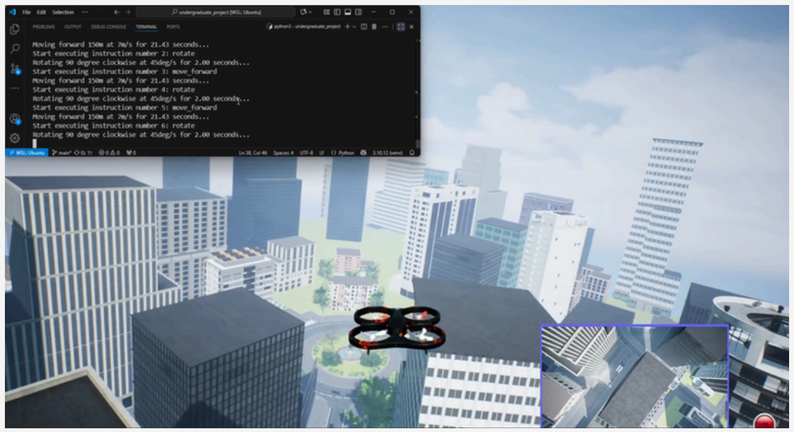
Result - Autonomous Flight

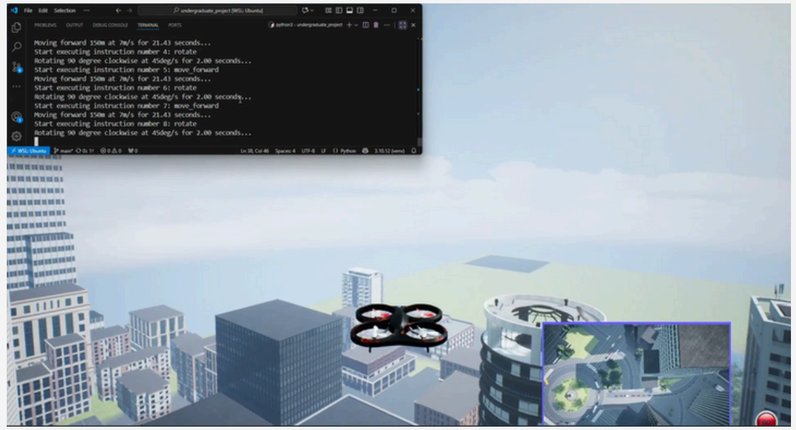
Navigation: “fly in a square that has a length of 150m”

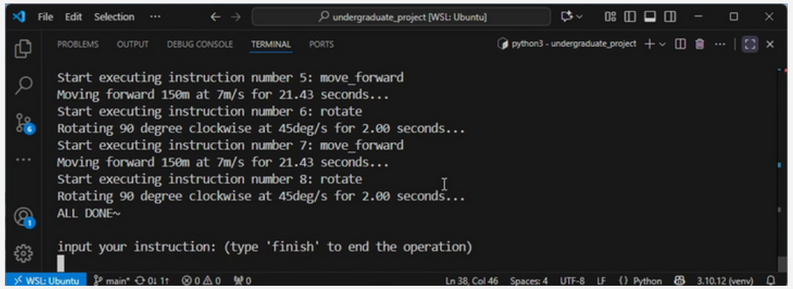








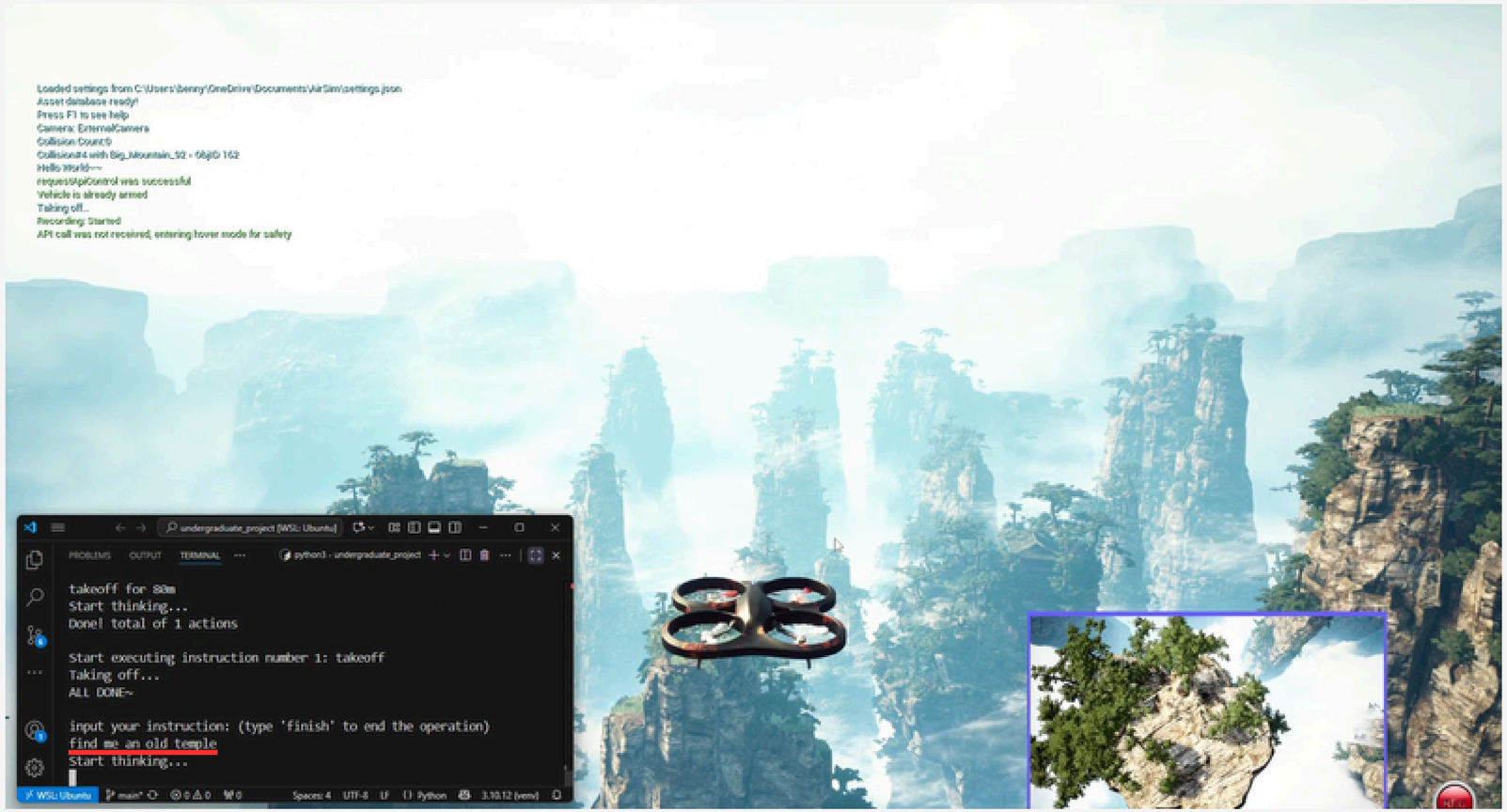




We can see it flew back to where it started

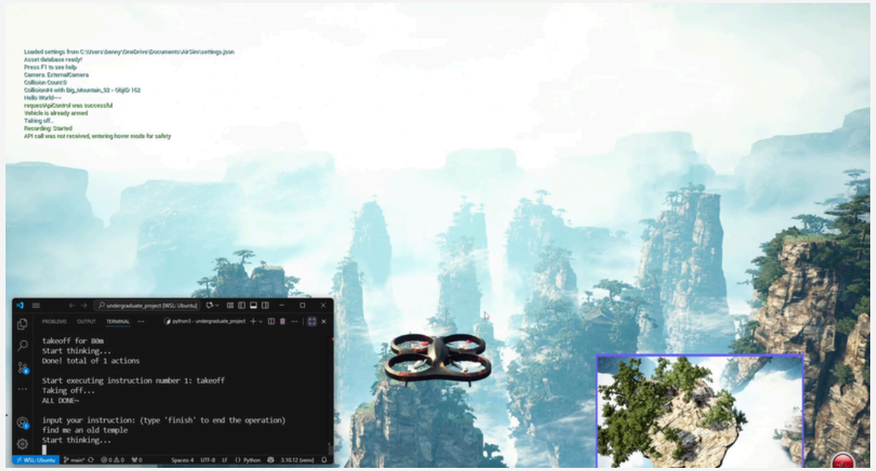
Result - Autonomous Flight

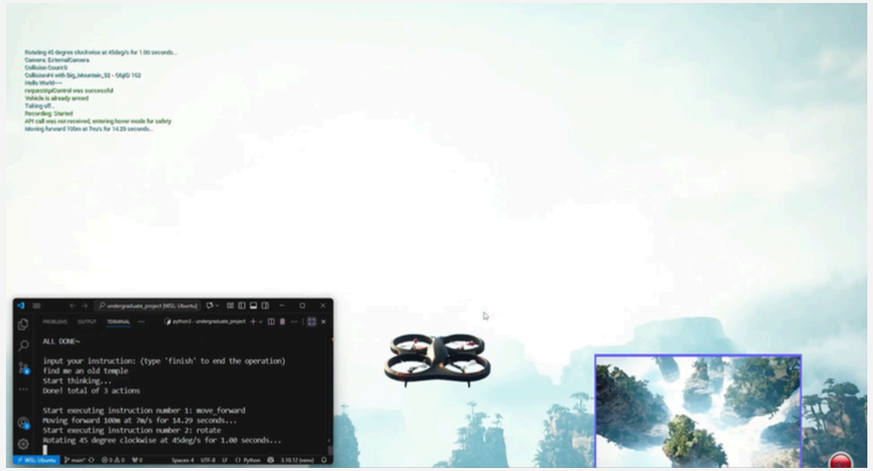
(Vision) Task: “find me an old temple”

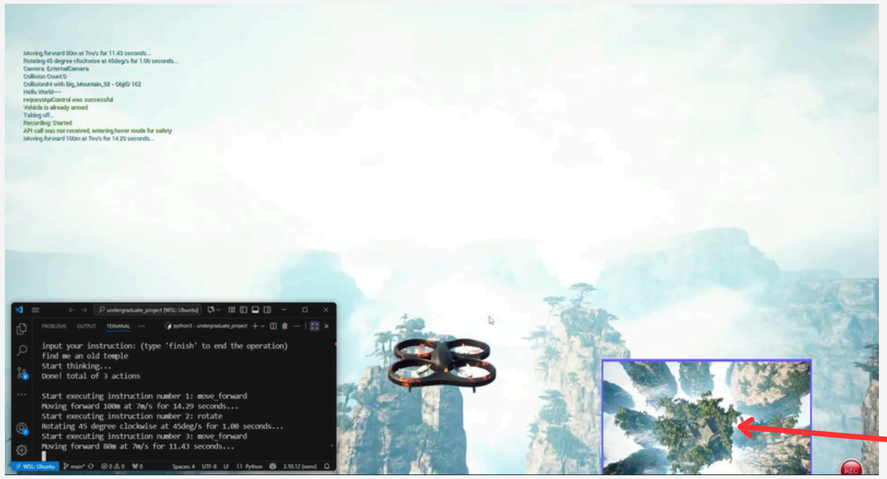


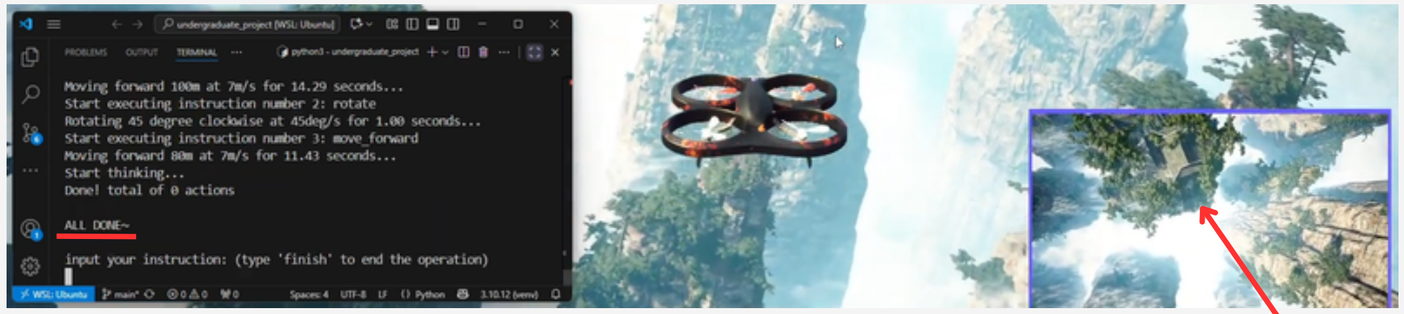
Result - Autonomous Flight

(Vision) Task: “find me an old temple”





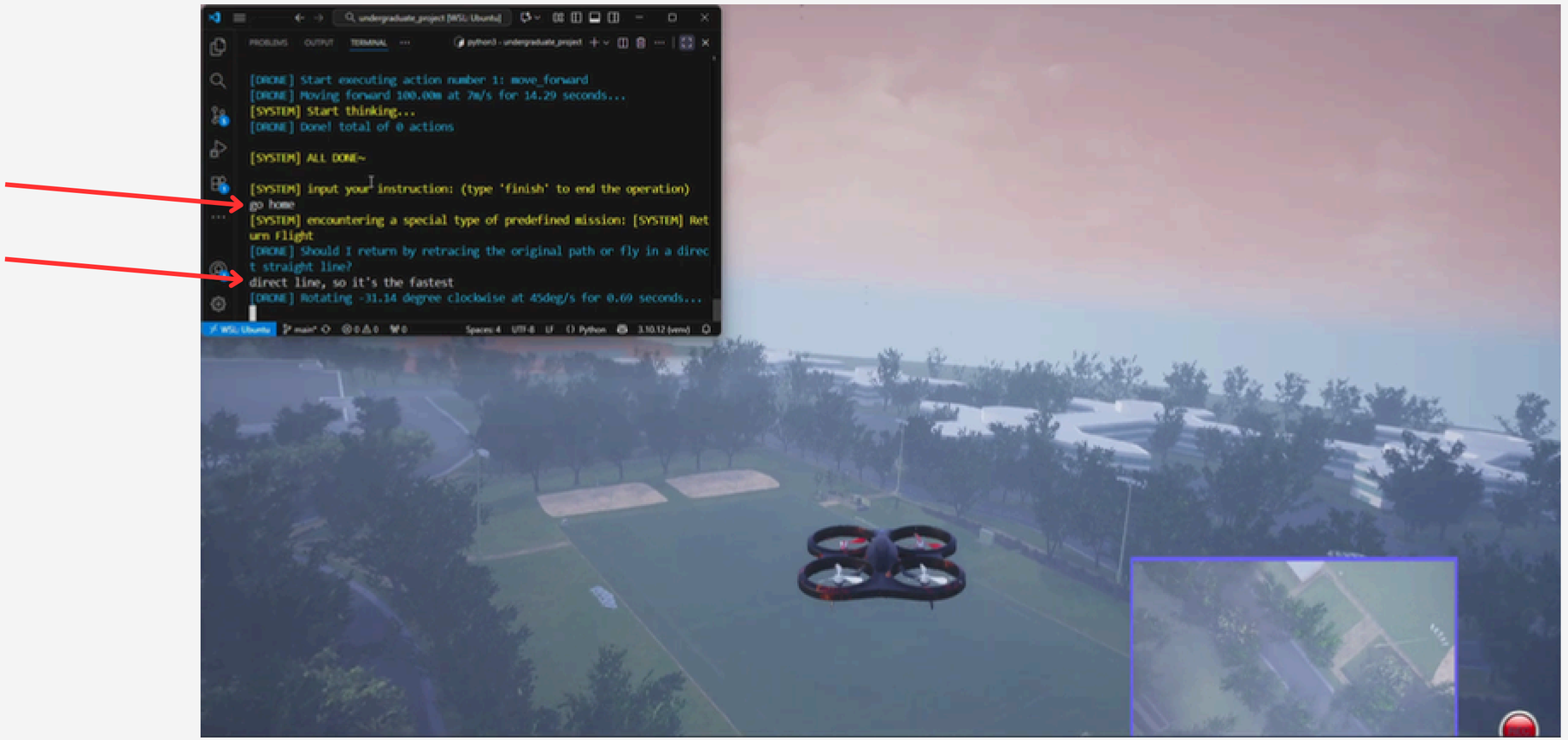




We can see it found a temple and stopped

Result - Autonomous Flight

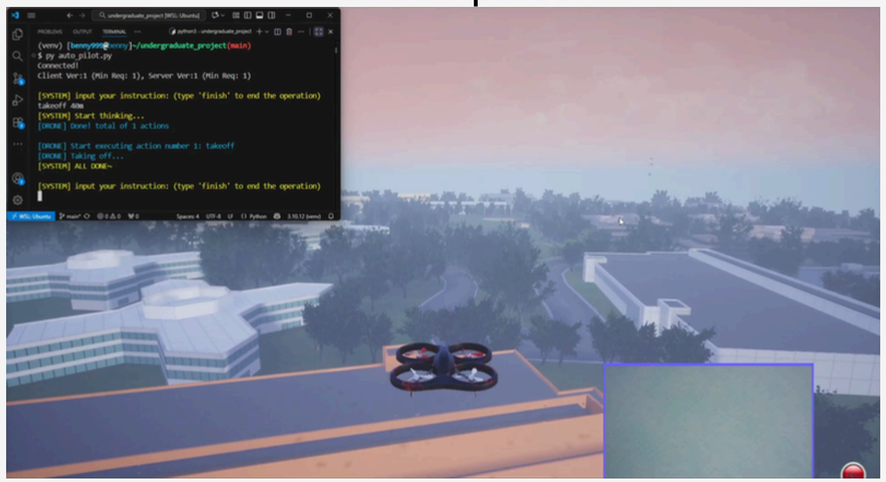
Predefined Mission: “Return Flight - direct” (“go home”)

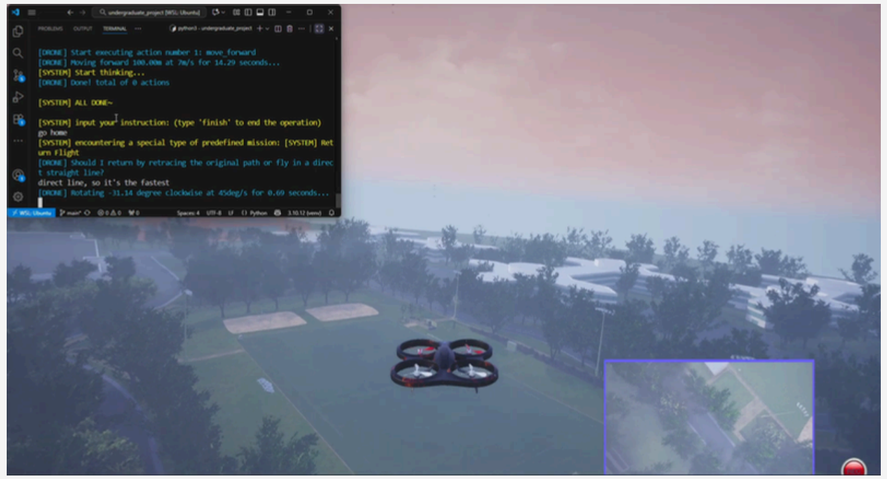


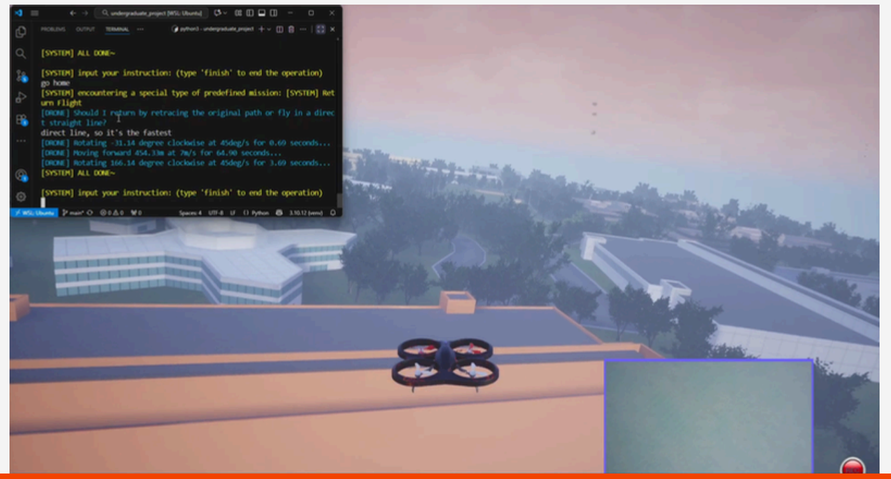
Result - Autonomous Flight

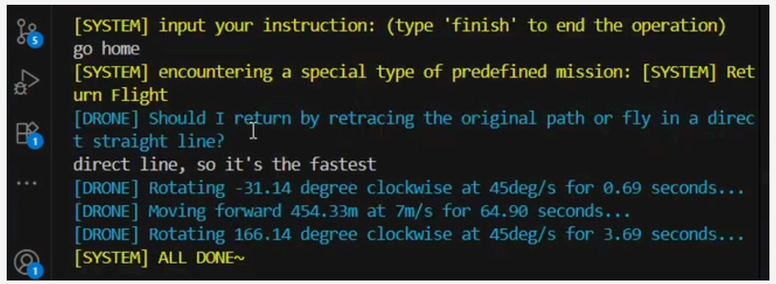
Predefined Mission: “Return Flight - direct” (“go home”)

Takeoff place





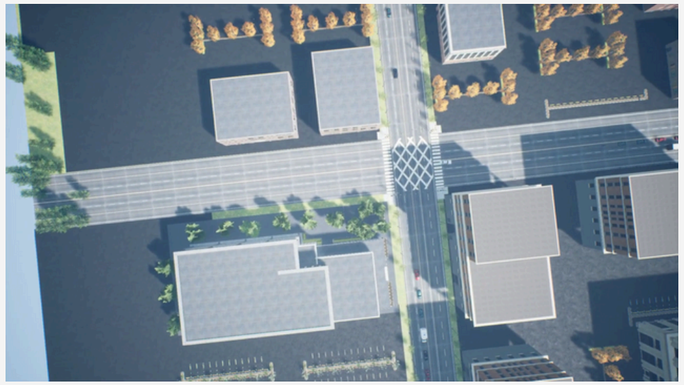


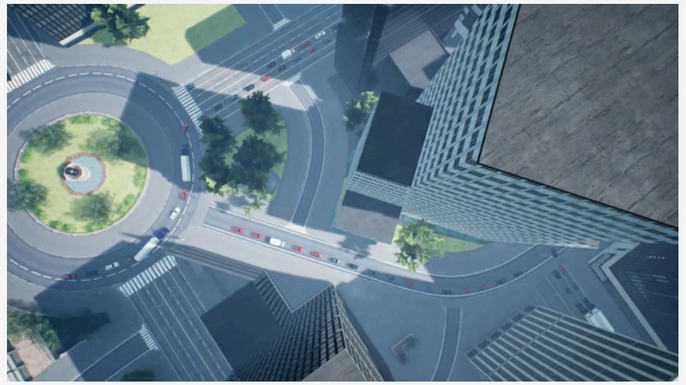


We can see it flew back to the takeoff place

Result - Flight Log in AirSim

Simulated city





Result - Flight Log

Simulated city

DRONE FLIGHT LOG

FLIGHT IDENTIFICATION

Date: 2025-09-22
Start Time: 10:00
End Time: 10:03
Total Duration: 2 min 20 sec

FLIGHT PURPOSE & OPERATIONS

Purpose of Flight: City Safety Monitoring
Type of Operation: Aerial Surveillance

LOCATION & ENVIRONMENT

Location Name/Description: Simulator city - urban downtown core with waterfront area (starts over undeveloped land, transitions through suburban commercial/residential, then dense urban, and ends near a coastal residential area)
GPS Coordinates (Takeoff): Open land in rural area
GPS Coordinates (Landing): Other place (not visible in the video)
Weather Conditions:

- Wind Speed: Appears calm (no visible wind effects on trees or water)
- Wind Direction: N/A
- Visibility: Excellent, clear conditions with sharp visibility
- Temperature: Not mentioned
- Cloud Cover: Clear skies
- Precipitation: None

FLIGHT PARAMETERS

Maximum Altitude: Not visible (continuously increasing, reaches high altitude over skyscrapers)
Maximum Distance: Not visible (covers a significant area of the simulated city)
Flight Pattern: ☑ Free Flight
Key Waypoints/Locations: Undeveloped grassy land, industrial facility, commercial buildings, residential houses, downtown intersection with roundabout, rail crossing with tram lines, downtown skyscrapers including some with rooftop solar panels, waterfront area with rocky shoreline, coastal area/beach, suburban residential neighborhood
Flight Path Summary: The drone begins over an open, undeveloped grassy area, then transitions over a suburban zone featuring commercial buildings and residential houses. Started over industrial area, moved through downtown core with multiple intersections and roundabouts, progressed to waterfront area with rocky shoreline and beach, ended in suburban residential zone. The camera maintains a high, top-down perspective throughout, gradually increasing altitude and revealing more of the simulated city with smooth transitions between locations.

CAMERA & RECORDING SETTINGS

Video Resolution: 1920x1080
Frame Rate: 20.0 fps
Recording Format: MP4

SAFETY CONSIDERATIONS

Obstacles Present: ☑ Trees ☑ Buildings ☑ Other: High-rise towers, tram lines, roundabouts, roads, water body
People in Area: ☑ Small Group (workers with hard hats observed, two figures on rooftop) - *Note: One log reported no people visible*
Emergency Landing Sites: Numerous open grassy areas, large parking lots, vacant lots, and plaza areas observed, especially in the earlier parts of the flight

INCIDENTS & OBSERVATIONS

Any Issues Encountered: Initial focus problems with blurry footage that quickly transitions to clear imagery
Wildlife Interactions: None observed
Signal Loss Events: None mentioned
Weather Changes: No changes observed; consistent clear, bright, sunny conditions throughout
Equipment Malfunctions: Initial focus/stability issues resolved quickly

NOTES & LESSONS LEARNED

Flight Performance: Smooth and stable flight with consistent movement, smooth transitions between locations, and excellent camera control
Footage Quality: High quality after initial focus adjustment - sharp, clear imagery with excellent detail of urban infrastructure, good exposure, and well-exposed footage
Areas for Improvement: Initial camera focus calibration could be improved
Future Considerations: Excellent documentation of urban traffic patterns, infrastructure layout, and city planning elements for safety monitoring purposes. The current flight provides a good overview of different urban and suburban zones. For future safety monitoring, specific areas of interest (e.g., high-traffic intersections, public gathering spaces, critical infrastructure) could be identified for more detailed, lower-altitude inspections. The flight path demonstrates capability for broad area coverage.

Very accurate overall

Future

Future

Problem: VLM’s vision ability

It’s challenging to successfully implement the predefined mission: Follow Path (at P.52).
The main issue is that VLM’s vision abilities are still lacking.

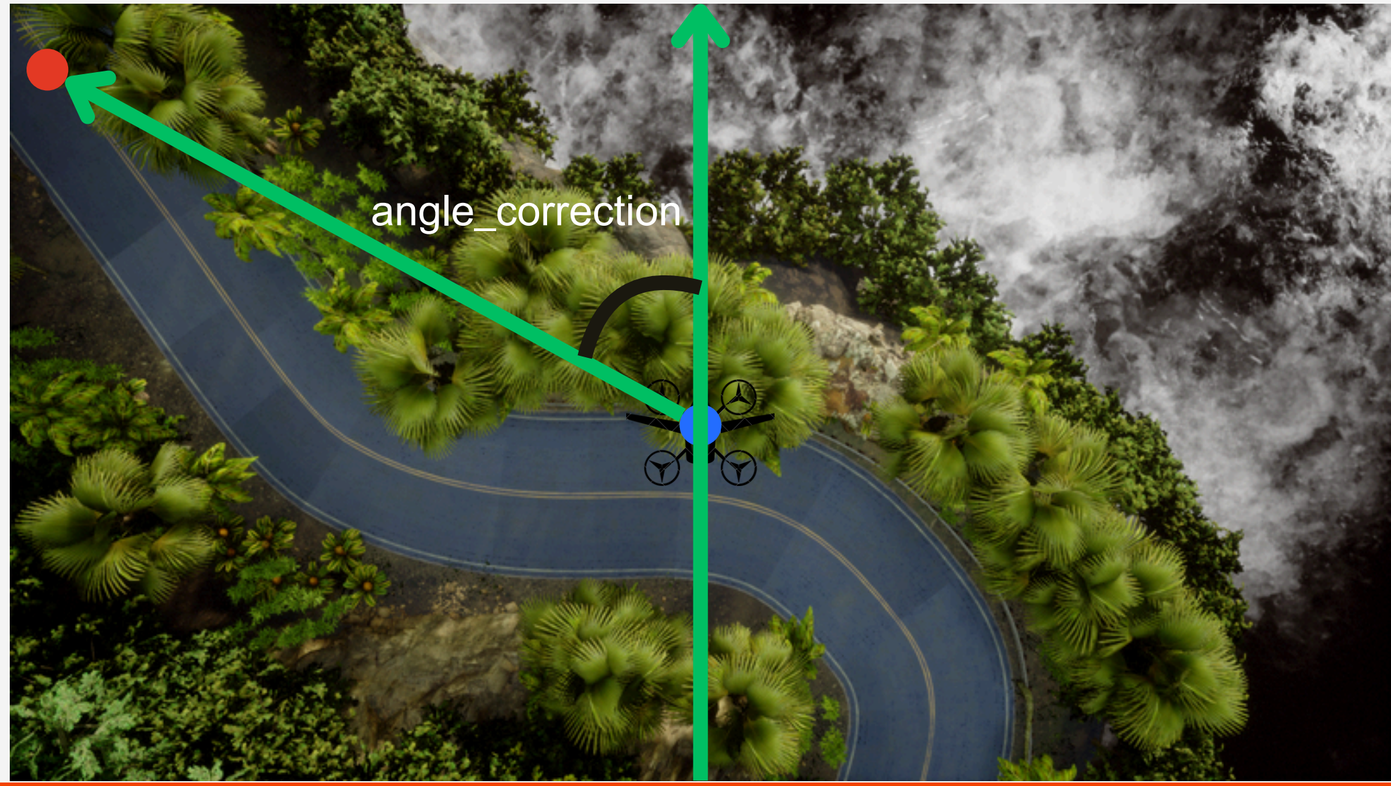
Future

Here’s a detailed experiment on these models’ vision ability. Testing if they can stably perform the angle correction:

YouTube link for live demo

actual data

In short, all of them failed to succeed.



End