

Supplementary Information for

Embodied Intelligence Enables Agentic Exploration in Microscopy

Gang Huang, Zhengyang Zhang, Songlin Zhuang, Xinghu Yu, Yang Wu, Zhan Li, Zhitai Liu,
Zhihui Lu, Mingsi Tong, Li Zhang, Huijun Gao

Corresponding author: tongms@hit.edu.cn, lizhang@cuhk.edu.hk, hjgao@hit.edu.cn.

The PDF file includes:

Supplementary Notes
Figs. S1 to S2
Tables S1 to S3

Other Supplementary Information for this manuscript include the following:

Videos S1 to S4

Supplementary Notes

Supplementary Note 1. Inter-Module Context Management

Executing long-horizon tasks in automated microscopy, such as “acquiring a global low-magnification image, analyzing cellular positions, and subsequently performing targeted high-magnification imaging”, necessitates multi-step decision-making and the coordination of multiple specialized modules. In this sequential workflow, the input for modules such as microscope control, image processing, and image analysis is often derived from the output of the preceding one. The requisite information extends beyond raw image data to include critical metadata, such as image dimensions, object counts, acquisition parameters, processing methods, and analysis results. A common approach of serially concatenating all contextual information into the system prompt is untenable for complex tasks. This method leads to a rapid accumulation of text, potentially exceeding the model context window limitations, and more critically, obscures essential information within a verbose and unstructured operational history. Therefore, a sophisticated mechanism is required to manage and transfer contextual information, ensuring both conciseness and the preservation of high-fidelity data.

To address this challenge, a structured information transfer protocol is proposed to achieve dynamic context engineering. This framework conceptualizes the LLM as the Decision-Maker and the underlying instrument control and analysis programs as Executors. An intermediary Information Interaction Module facilitates communication between them, adhering to a defined protocol. This module comprises two key components: 1) Context Extractor. This component is responsible for parsing the textual output generated by the Decision-Maker to identify and isolate critical information that must be persisted. It effectively tells the Executor what to save from the current operation. 2) State Formatter. This component organizes the diverse information saved by the Executors into a structured, consolidated format. This formatted state summary is then presented back to the Decision-Maker to inform the next decision cycle. Furthermore, an incremental update mechanism is adopted: replacing the complete rewrites with local edits, which significantly reduces latency and computational costs.

The workflow is implemented within our system through a centralized StorageManager component that enables cross-module data sharing. When a new low-level policy decision is required, the Context Extractor first distills key information from the LLM directive, tailored to the requirements of the downstream module. Subsequently, the State Formatter applies task-specific rules to dynamically structure and store this information. This process includes managing file I/O operations: files are saved to designated directories, and a corresponding metadata entry is created in the StorageManager. This entry, which forms part of a comprehensive metadata list, contains structured information about the file rather than the raw data itself. The StorageManager is equipped with file-access functions that allow for the direct retrieval of any file via its identifier from the metadata.

Specifically, modules share structured data via metadata dictionaries. The core metadata structure for data generated by the Microscope Control, Image Processing, and Image Analysis modules includes: a unique identifier for the data file; a composite field containing essential parameters (e.g., fluorescence channels, pixel size, magnification) and an LLM-generated summary of the file’s purpose and context; the name of the module that generated the file; and the format of the file (e.g., TIFF, CSV). For instance, in a task involving imaging scattered organoids within a culture dish, the system first acquires a global scan. The corresponding metadata for this initial image is stored, capturing its overarching context.

```
{'brightfield_2x2cm.ome.tif': {'filename': 'brightfield_2x2cm.ome.tif',  
'description': 'channel_names: [(128, 128, 128)], pixel_size: 1.62,  
magnification: 4', 'created_by': 'microscope', 'file_type': 'ome-tiff'}}
```

Following this, an organoid localization operation is performed on the global image. Based on the characteristics identified by the Image Analysis module, the contextual history is updated with new, refined information.

```
{'brightfield_2x2cm.ome.tif': {'filename': 'brightfield_2x2cm.ome.tif',  
'description': 'channel_names: [(128, 128, 128)], pixel_size: 1.62,  
magnification: 4', 'created_by': 'microscope', 'file_type': 'ome-tiff'},  
'organoid_locations_list.json': {'filename': 'organoid_locations_list.json',  
'description': 'Organoid positions detected in 2x2 cm brightfield image based on  
morphological features', 'created_by': 'analysis_platform', 'file_type': 'json'}}
```

This updated context is subsequently fed into the next decision-making cycle to guide detailed imaging of individual organoids. In another scenario, where the objective is to acquire images of organoids distributed throughout an entire droplet and compute their density, the system first performs a multi-layer Z-stack acquisition, followed by depth-of-field extension. Throughout this process, the raw microscope images and the resulting extended-depth-of-field image are sequentially saved. Their associated metadata is then passed to the analysis module to facilitate density calculation.

```
{'cell_3D_structure.ome.tif': {'filename': 'cell_3D_structure.ome.tif',  
'description': 'channel_names: [(128, 128, 128)], pixel_size: 1.62,  
magnification: 4', 'created_by': 'microscope', 'file_type': 'ome-tiff'},  
'extended_depth_cell_3D_structure.ome.tif': {'filename':  
'extended_depth_cell_3D_structure.ome.tif', 'description': 'Extended depth of  
field image generated via maximum intensity Z-projection', 'created_by':  
'analysis_platform', 'file_type': 'ome-tiff'}}
```

In conclusion, at each new decision-making step, the system retrieves this curated metadata list from the StorageManager and injects it into the LLM's prompt as supplementary context. This structured format enables the LLM to autonomously reference the history of operations and locate the specific information required for the current task. The LLM then communicates its new decisions and any information to be persisted through designated function calls, which in turn update the StorageManager. This completes the robust cycle of inter-module communication, ensuring efficient and context-aware execution of complex microscopy tasks.

Supplementary Note 2. An Interface for Large Language Model-Driven Microscopy

The management of a diverse suite of microscopy tools is predicated upon a framework that enables an LLM to not only identify individual instruments but also to comprehend the semantic context and scientific purpose of their application, transcending mere mechanical invocation. To achieve this, our approach relies on the synergistic use of two key components: a tool configuration manifest and a corresponding tool class.

Within this architecture, the class name programmatically designates the controllable instrument or module. Its method interfaces must be defined with clarity and precision, maintaining a strict separation between private implementation details and the public Application Programming Interface (API). This ensures that all instrument control is exclusively and safely managed through this public API. In our system, each tool class is architecturally defined as an independent, modular component. Consequently, a single class often encapsulates multiple distinct executable functions (e.g., *acquire_image*, *move_stage*). Crucially, the definition for each tool must be accompanied by comprehensive metadata, including a detailed functional description, specifications for inputs and outputs, and any operational caveats or prerequisites. This metadata can be specified either within the code as docstrings or in the tool configuration manifest. For instance, the tool class for a FRAP experiment would encapsulate methods for photobleaching, time-lapse imaging, and data analysis, each with its own detailed description.

```
import random
from typing import List, Tuple
import math

class Frap:
    def __init__(self):

    def laser_on(self, power: float, duration: float) -> None:
        """
        Activate laser for precise cell manipulation
        Args:
            power: Laser intensity (percentage of max power, 0.0-100.0)
            duration: Exposure time in milliseconds (ms)
        Raises:
            ValueError: If power < 0 or > 100, or duration <= 0
        """

    def laser_off(self) -> None:
        """
        Immediately deactivate laser emission
        """

    @staticmethod
    def laser_position(x: int, y: int) -> None:
        """
        Set laser focal point coordinates
        Args:
            x: X-axis position (microns)
            y: Y-axis position (microns)
        """

    @staticmethod
    def cell_detection() -> dict:
        """
        Detect and return the position of target cell
        Returns:
            Dictionary containing cell coordinates:
            - 'x': X-axis position (microns)
            - 'y': Y-axis position (microns)
        """
```

```

"""
@staticmethod
def cell_contour_extraction() -> dict:
"""
    Extract cell membrane contour from current image
    Returns:
        Dictionary containing contour data:
        - 'points': List of (x,y) tuples (microns)
        - 'area': Cell area in square microns ( $\mu\text{m}^2$ )
        - 'perimeter': Cell perimeter in microns ( $\mu\text{m}$ )
    Note:
        Returns empty dict if no cell detected
"""

```

The tool registration process begins with the tool configuration manifest. This manifest, structured as a dictionary adhering to a predefined schema, specifies the tool and all necessary settings for its execution. This manifest is then processed by a Tool Manager, which automates the registration protocol. Upon successful registration, the tool is dynamically integrated into the system's central *module_map*, obviating the need for any further manual configuration. At this point, the contents of the tool configuration manifest have been updated as follows:

```

frap_methods = [
    "laser_on",
    "laser_off",
    "laser_position",
    "cell_detection",
    "cell_contour_extraction"]

Frap_vars = {
    k: getattr(env_frap, k)
    for k in frap_methods}

module_map = {
'Microscope Operation Platform': prompt_olympus,
'Cell Segmentation Platform': prompt_cellpose,
'Image Analysis Platform': prompt_imagej,
'Micromanipulation system': prompt_micro,
'FRAP': prompt_frap}

```

A pivotal step in this process is the automated generation of a comprehensive prompt tailored for the LLM. This prompt is a composite artifact that encapsulates: 1) formal function definitions derived from the public API; 2) illustrative usage examples, and; 3) optionally, supplementary domain-specific knowledge to contextualize the tool's use. The fully formed prompt is then serialized into a dedicated configuration file (The corresponding files generated for the FRAP module mentioned above are as follows). Subsequently, the main control program dynamically imports the corresponding function module, automatically introspecting its available methods and variables. At this point, the autonomous agent is fully equipped and ready for execution.

```

prompt_Frap = '''
# Role and Objectives

```

Role: A professional assistant specialized in generating Python code for controlling Frap systems based on user instructions.

Core Objective: Ensure the generated code is secure and complies with hardware constraints.

```
# Behavioral Constraints
- Language use: English
- Determine parameters based on Behavioral Constraints and Current
environment
## Hardware security control
- All motion and imaging commands must include parameter verification.
## Context-Aware
- Fully utilize user-provided file information to avoid assuming non-existent
files/parameters
## Decision-Making Mechanism
- No assumptions are allowed.
- Prioritize using the provided API functions to complete tasks. Carefully
read the API function definitions before answering to ensure that the tasks can
be completed and comply with the function definitions.
- Saving mechanism: Use the provided functions to read and save files.
- Execute each instruction sequentially and completely without skipping,
merging, or reordering.
# API function
def laser_on(power: float, duration: float) -> None:
    """
    Activate laser for precise cell manipulation

    Args:
        power: Laser intensity (percentage of max power, 0.0-100.0)
        duration: Exposure time in milliseconds (ms)

    Raises:
        ValueError: If power < 0 or > 100, or duration <= 0
    """
    ...(Omitted)

# Usage Example
# Example input
Demonstrate laser control and cell detection workflow including positioning
the laser, detecting a cell, extracting its contour, and activating the laser
with specific power and duration.

# Example output
...(Omitted) '''
```

For the above steps, we further describe the specific implementation. To operationalize this registration pipeline, we have developed a specialized utility named the *LLM Agent*. This agent is responsible for parsing and deconstructing the tool classes and their associated configuration manifests. Its core workflow comprises the following automated stages:

1. LLM Configuration: Initializes the connection and parameters for the designated LLM.
2. Code Analysis: Systematically scans the target source code directory to discover and extract class definitions, public method signatures, and associated docstrings.

3. Documentation Generation: For any public methods lacking sufficient documentation, it leverages the LLM to automatically generate descriptive docstrings that explain their functionality.

4. Example Synthesis: Generates practical, context-aware usage examples and constructs high-level task planning templates based on the tool’s capabilities.

5. File Output: Produces a complete, machine-readable API documentation file and updates the system’s central task manager with the newly registered tool’s profile.

Through this automated pipeline, the *LLM Agent* leverages the public method interfaces and their accompanying annotations (docstrings) to generate specific, context-rich usage examples. These examples are then integrated into a standardized template to produce the final, complete prompt file that instructs the LLM. Furthermore, the agent synthesizes a concise summary of the tool’s capabilities from its public methods. This summary is then registered with the system’s task allocation module, enabling the higher-level agent to intelligently delegate experimental tasks to the most appropriate microscopy tool.

Supplementary Note 3. Execution Workflow of the EIMS

This section provides a comprehensive, step-by-step walkthrough of the EIMS execution process, illustrating how a high-level, natural-language user request is parsed, decomposed, and executed by the hierarchical multi-agent system.

We categorize the model inputs into two distinct paradigms tailored to the dynamic demands of users: an autonomous exploration model for investigative scenarios where imaging targets and analytical objectives remain unspecified, and a copilot model for goal-directed tasks with predefined experimental outcomes. Analogous to deepresearch agents, the autonomous exploration paradigm takes as input a core research question (e.g., the effects of paclitaxel on tumor organoids), retrieves relevant data from curated repositories, synthesizes the specific imaging phenotypic alterations and critical acquisition considerations, and translates these insights into a structured research plan encompassing optimized microscopy acquisition parameters and quantitative image analysis metrics. This integrated information is then processed via a dedicated summarization module to generate a standardized experimental protocol—aligning the input format with that required for the copilot model. In contrast, the copilot paradigm is initiated by a user-specified request composed of two fundamental components. The process is initiated with a user-defined request comprising two key components: the Imaging Target and the Imaging Task. For this example, we consider a typical organoid imaging experiment:

User Request:

Imaging Target: Organoids;

Imaging Task: Detect organoids at the current position using the low-magnification brightfield mode and record their coordinates; switch to the 20x objective, capture images of the blue, green and red fluorescent channels in the area of the second organoid's position, and merge the images.

1. High-Level Planning and Interactive Refinement

Upon receiving the request, the high-level policy, orchestrated by the LLM, first parses the natural language prompt. It identifies a semantic ambiguity in the instruction “low-magnification brightfield mode”. The term lacks a precise hardware correlate. To resolve this, the self-correction module is triggered, initiating a query to clarify the user’s intent:

Inference Process:

The generated plan contains an ambiguous directive, as the specific objective lens for 'low-magnification' is not specified.

System Query:

The term 'low-magnification' is ambiguous. Should the 4x or 10x objective be used?

Based on established laboratory protocols for organoid screening, where a 4× objective is standard for obtaining a wide field of view, the user is provided with a clarifying directive:

User Directive:

Use 4x objective

With this ambiguity resolved, the system re-evaluates the plan to ensure semantic consistency and generates a finalized high-level strategy. This strategy decomposes the overarching goal into a sequence of discrete subtasks, assigning each to a specialized agent (e.g., Microscope Control, Image Analysis). Within each subtask, operations are further broken down into parameter settings and action primitives.

```
[{'subtask_index': 1, 'module': 'Microscope Operation Platform', 'command':  
'Parameter Setting: Set the currently used objective lens to 4x and set the  
filter set to brightfield mode;  
#Auxiliary Operation: Adjust the light source brightness to an appropriate level  
for brightfield imaging; perform auto-focus at the current position;  
#Image Automatic Acquisition Parameter Setting: Configure the XY position  
parameter to the current field of view; set the filter set to brightfield mode  
and the exposure parameter to the current value; do not configure Z-axis stack or  
time parameters;  
#Image Automatic Acquisition: Perform automatic image acquisition using the  
configured parameters to obtain a brightfield image of the current field of  
view'},  
{'subtask_index': 5, 'module': 'Image Analysis Platform', 'command': 'Image  
Import: Import the acquired brightfield image (OME-TIFF format);  
#Target Detection: Detect organoid regions in the imported brightfield image and  
save the detection results as a JSON file containing the coordinates of each  
organoid'},  
{'subtask_index': 7, 'module': 'Microscope Operation Platform', 'command':  
"Parameter Setting: Set the currently used objective lens to 20x;  
#Target Position Loading: Load the coordinates of detected organoid regions from  
the JSON file;  
#Position Control: Move the stage to the area corresponding to the second  
organoid's coordinates;  
#Parameter Setting: Set the filter set to blue fluorescence mode, configure the  
camera exposure time for blue fluorescence imaging, and set the light source  
brightness to 0;
```

```

#Auxiliary Operation: Perform auto-focus at the second organoid's position to
ensure accurate focus for fluorescence imaging;
#Image Automatic Acquisition Parameter Setting: Configure the filter set to blue
fluorescence and set the corresponding exposure parameter; set the XY position
parameter to the second organoid's coordinates; do not configure Z-axis stack or
time parameters;
#Image Automatic Acquisition: Perform automatic image acquisition using the
configured parameters to capture the blue fluorescence channel image;
#Parameter Setting: Set the filter set to red fluorescence mode, configure the
camera exposure time for red fluorescence imaging, and ensure the light source
brightness is 0;
#Image Automatic Acquisition Parameter Setting: Configure the filter set to red
fluorescence and set the corresponding exposure parameter; set the XY position
parameter to the second organoid's coordinates; do not configure Z-axis stack or
time parameters;
#Image Automatic Acquisition: Perform automatic image acquisition using the
configured parameters to capture the red fluorescence channel image"},
{'subtask_index': 17, 'module': 'Image Analysis Platform', 'command': 'Image
Import: Import the acquired blue and red fluorescence channel images (OME-TIFF
format);
#Image Merging: Merge the imported blue and red fluorescence channel images into
a composite image'}}]

```

Notably, to enhance operational robustness and prevent the LLM from generating potentially erroneous, low-level command sequences for routine procedures, the Microscope Control Agent is constrained to utilize a predefined set of encapsulated functions. These functions correspond to four primary categories: Parameter Setting (direct hardware parameter control), Auxiliary Operation (composite routines such as auto-focusing and automated brightness adjustment, plus access to stored data such as well-plate coordinates), Image Automatic Acquisition Parameter Setting (configuration for automated imaging sequences), and Image Automatic Acquisition (initiation of the configured capture).

2. Subtask Execution: Low-Magnification Scan

The first subtask is dispatched to the Microscope Control Agent. The input provided to the agent includes the current system state and the specific actions to be performed. At this initial stage, the shared document memory is empty.

```

# Input to Microscope Control Agent:
# Saved documents:
{}
# Current environment: current xy_position: (2500, 2500), z_position:4900,
_exposure_time:10, objective:1-UPLFLN4XPH, dichroic:1-NONE, brightness:70
# Task Sequence:
# Parameter Setting: Set the currently used objective lens to 4x and set the
filter set to brightfield mode;

```

```

# Auxiliary Operation: Adjust the light source brightness to an appropriate level
for brightfield imaging; perform auto-focus at the current position;
# Image Automatic Acquisition Parameter Setting: Configure the XY position
parameter to the current field of view; set the filter set to brightfield mode
and the exposure parameter to the current value; do not configure Z-axis stack or
time parameters;
# Image Automatic Acquisition: Perform automatic image acquisition using the
configured parameters to obtain a brightfield image of the current field of view.

```

The low-level policy within the agent translates this structured input into a sequence of hardware-specific API calls. This layer of abstraction ensures that the LLM's high-level plan is converted into precise, executable commands without needing to manage the complexities of device dynamics. In some cases, we refer to this step as the intermediate layer, while the low-level policy refers to the specific control algorithms that execute specific actions. However, in the control of microscopes, as discussed in the main text, we generally do not need to consider complex dynamics, and thus the control algorithms for the execution of such actions can be ignored. Therefore, we tend to call the model that converts abstract tasks into specific API calls the low-level policy. Based on above input information, the task sequence is translated into the corresponding execution program.

```

say("[INFO] Starting parameter setup: 4x objective and brightfield mode")
target_objective = '1-UPLFLN4XPH'
current_objective = get_objective()
if current_objective != target_objective:
    set_objective(target_objective)
    say(f"[INFO] Objective lens set to 4x (label: {target_objective})")
else:
    say(f"[INFO] Objective lens is already 4x (label: {target_objective}), no
change needed")
target_channel = '1-NONE'
current_channel = get_channel()
if current_channel != target_channel:
    set_channel(target_channel)
    say(f"[INFO] Filter set to brightfield mode (channel: {target_channel})")
else:
say(f"[INFO] Filter is already in brightfield mode (channel: {target_channel}),
no change needed")

say("[INFO] Automatically adjusting brightness for brightfield imaging")
optimal_brightness = perform_autobrightness()
set_brightness(optimal_brightness)
say(f"[INFO] Brightfield brightness set to optimal value: {optimal_brightness}")
say("[INFO] Performing auto-focus for brightfield imaging")
optimal_z = perform Autofocus()
set_z_position(optimal_z)

```

```

say(f"[INFO] Brightfield auto-focus completed, Z-axis position: {optimal_z} μm")

say("[INFO] Configuring brightfield image acquisition parameters")
current_x, current_y = get_x_y_position()
say(f"[INFO] Current XY position: ({current_x}, {current_y}) μm")
add_acquisition_position(name="brightfield_4x", x=current_x, y=current_y,
width=None, height=None)
current_exposure = get_exposure()
add_channels(channel=target_channel, exposure=current_exposure)
say(f"[INFO] Brightfield acquisition parameters configured: channel
{target_channel}, exposure {current_exposure} ms")

say("[INFO] Starting 4x brightfield image acquisition")
run_acquisition()
say("[INFO] 4x brightfield image acquisition completed")

```

3. Subtask Execution: Image Analysis and Target Identification

Upon successful acquisition of the brightfield image, its metadata is stored in the shared memory. The high-level planner then dispatches the next subtask—target detection—to the Image Analysis Agent.

```

# Input to Image Analysis Agent:
# Saved documents:
{'brightfield_4x.ome.tif': {'filename': 'brightfield_4x.ome.tif', 'description':
'channel_names: [(128, 128, 128)], pixel_size: 1.62, magnification: 4',
'created_by': 'microscope', 'file_type': 'ome-tiff'}}
# Task Sequence:
# Image Import: Import the acquired brightfield image (OME-TIFF format);
# Target Detection: Detect organoid regions in the imported brightfield image and
save the detection results as a JSON file containing the coordinates of each
organoid.

```

The Image Analysis Agent executes this task by loading the specified image file and applying a pretrained computer vision model to identify and segment organoids. Since the previous step collected the images and placed them in the memory module, they are now input into the current module as new data. The description of the image includes the file path, metadata description, channels, pixel size, magnification, file format, etc. This facilitates the image processing module in handling the files based on their characteristics. The center coordinates of each detected object are calculated and stored in a new JSON file, which is then added to the shared document memory. The specific implementation steps are as follows:

```

fiji_initialize()

# Load the brightfield image
input_file = "brightfield_4x.ome.tif"

```

```

say("[ACTION] Loading brightfield image: " + input_file)
image = load_image(input_file)

# Detect organoid regions in the brightfield image
say("[ACTION] Detecting organoid regions in the brightfield image")
organoid_positions = analysis_platform_find_organoid_position(image, "Detected
organoid regions in brightfield image (4x magnification)")
say("[ACTION] Detection results saved as JSON file")

fiji_shutdown()

```

4. Subtask Execution: High-Magnification Multi-Channel Imaging

With the organoid locations identified, the workflow proceeds to the high-magnification imaging phase. The Microscope Control Agent is invoked again, this time with the location data from the previous step.

```

# Input to Microscope Control Agent:
# Saved documents:
'organoid_locations_list.json': {'filename': 'organoid_locations_list.json',
'description': 'Detected organoid regions in brightfield image (4x
magnification)', 'created_by': 'analysis_platform', 'file_type': 'json'}}
# Current environment: current xy_position:(2500,
2500),_z_position:4899.0448064534, _exposure_time:10,_objective:1-
UPLFLN4XPH,_dichroic:1-NONE,_brightness:59
#Parameter Setting: Set the currently used objective lens to 20x;
# Task Sequence:
# Target Position Loading: Load the coordinates of detected organoid regions from
the JSON file;
# Position Control: Move the stage to the area corresponding to the second
organoid's coordinates;
# Parameter Setting: Set the filter set to blue fluorescence mode, configure the
camera exposure time for blue fluorescence imaging, and set the light source
brightness to 0;
# Auxiliary Operation: Perform auto-focus at the second organoid's position to
ensure accurate focus for fluorescence imaging;
# Image Automatic Acquisition Parameter Setting: Configure the filter set to blue
fluorescence and set the corresponding exposure parameter; set the XY position
parameter to the second organoid's coordinates; do not configure Z-axis stack or
time parameters;
# Image Automatic Acquisition: Perform automatic image acquisition using the
configured parameters to capture the blue fluorescence channel image;
# Parameter Setting: Set the filter set to red fluorescence mode, configure the
camera exposure time for red fluorescence imaging, and ensure the light source
brightness is 0;

```

```
# Image Automatic Acquisition Parameter Setting: Configure the filter set to red
fluorescence and set the corresponding exposure parameter; set the XY position
parameter to the second organoid's coordinates; do not configure Z-axis stack or
time parameters;
# Image Automatic Acquisition: Perform automatic image acquisition using the
configured parameters to capture the red fluorescence channel image.
```

Following this detailed plan, the low-level policy executes a series of actions: commanding the objective turret to switch to 20×, parsing the JSON file to retrieve the target coordinates, instructing the motorized stage to move to the precise XY position, and then systematically cycling through the specified filter cubes and exposure settings to capture each fluorescence channel. An autofocus routine is performed before the first fluorescence capture to correct for any focal drift. The resulting multi-channel image data is saved as distinct OME-TIFF files. The detailed execution steps are as follows:

```
say("[INFO] Setting objective lens to 20x for organoid fluorescence imaging")
target_objective = '3-LUCPLFLN20XRC'
current_objective = get_objective()
if current_objective != target_objective:
    set_objective(target_objective)
    say(f"[INFO] Objective lens set to 20x (label: {target_objective})")
else:
    say(f"[INFO] Objective lens is already 20x (label: {target_objective}), no
change needed")

say("[INFO] Loading organoid region coordinates from JSON file")
organoid_json_filename = "organoid_locations_list.json"
say(f"[ACTION] Loading organoid locations from file {organoid_json_filename}")
organoid_bounding_boxes = load_target_locations(organoid_json_filename)
if len(organoid_bounding_boxes) < 2:
    say("[ERROR] Less than two organoid regions found in the file. Aborting
operation.")
else:
    say(f"[INFO] Successfully loaded {len(organoid_bounding_boxes)} organoid
region bounding boxes")
    second_organoid = organoid_bounding_boxes[1]
    x, y, width, height = second_organoid
    say(f"[ACTION] Moving stage to second organoid region: X={x} μm, Y={y} μm")
    if 0 <= x <= 500000 and 0 <= y <= 500000:
        set_x_y_position(x, y)
        say(f"[INFO] Stage moved to position X={x} μm, Y={y} μm")
    else:
        say("[ERROR] Target XY position out of hardware range. Aborting
operation.")
```

```

    say("[INFO] Setting filter set to blue fluorescence mode and configuring
imaging parameters")
    target_channel_blue = '2-U-FUNA'
    current_channel = get_channel()
    if current_channel != target_channel_blue:
        set_channel(target_channel_blue)
        say(f"[INFO] Filter set to blue fluorescence mode (channel:
{target_channel_blue})")
    else:
        say(f"[INFO] Filter is already in blue fluorescence mode (channel:
{target_channel_blue}), no change needed")
    target_exposure_blue = 100.0
    set_exposure(target_exposure_blue)
    say(f"[INFO] Camera exposure time set to {target_exposure_blue} ms for blue
fluorescence imaging")
    set_brightness(0)
    say("[INFO] Light source brightness set to 0 for fluorescence imaging")

    say("[INFO] Performing auto-focus at second organoid position for blue
fluorescence imaging")
    optimal_z_blue = perform Autofocus()
    set_z_position(optimal_z_blue)
    say(f"[INFO] Auto-focus complete, Z position set to {optimal_z_blue} µm")

    say("[INFO] Configuring automatic acquisition parameters for blue
fluorescence channel")
    add_acquisition_position(name="organoid2_blue", x=x, y=y, width=width,
height=height)
    add_channels(channel=target_channel_blue, exposure=target_exposure_blue)
    say(f"[INFO] Acquisition parameters set: channel {target_channel_blue},
exposure {target_exposure_blue} ms, position X={x}, Y={y}, size {width}x{height}
µm")

    say("[INFO] Starting automatic image acquisition for blue fluorescence
channel")
    run_acquisition()
    say("[INFO] Blue fluorescence channel image acquisition completed")

    say("[INFO] Setting filter set to red fluorescence mode and configuring
imaging parameters")
    target_channel_red = '4-U-FGNA'
    current_channel = get_channel()
    if current_channel != target_channel_red:
        set_channel(target_channel_red)

```

```

    say(f"[INFO] Filter set to red fluorescence mode (channel:
{target_channel_red})")
    else:
        say(f"[INFO] Filter is already in red fluorescence mode (channel:
{target_channel_red}), no change needed")
        target_exposure_red = 100.0
        set_exposure(target_exposure_red)
        say(f"[INFO] Camera exposure time set to {target_exposure_red} ms for red
fluorescence imaging")
        set_brightness(0)
        say("[INFO] Light source brightness set to 0 for fluorescence imaging")

    say("[INFO] Configuring automatic acquisition parameters for red fluorescence
channel")
    add_acquisition_position(name="organoid2_red", x=x, y=y, width=width,
height=height)
    add_channels(channel=target_channel_red, exposure=target_exposure_red)
    say(f"[INFO] Acquisition parameters set: channel {target_channel_red},
exposure {target_exposure_red} ms, position X={x}, Y={y}, size {width}x{height}
µm")

    say("[INFO] Starting automatic image acquisition for red fluorescence
channel")
    run_acquisition()
    say("[INFO] Red fluorescence channel image acquisition completed")

```

5. Subtask Execution: Image Processing

Finally, the acquired fluorescence images are passed back to the Image Analysis Agent for post-processing, as dictated by the initial user request.

```

# Input to Image Analysis Agent:
# Saved documents:
{'organoid2_blue.ome.tif': {'filename': 'organoid2_blue.ome.tif', 'description':
'channel_names: [(0, 0, 255)], pixel_size: 0.32, magnification: 20',
'created_by': 'microscope', 'file_type': 'ome-tiff'}, 'organoid2_red.ome.tif':
{'filename': 'organoid2_red.ome.tif', 'description': 'channel_names: [(255, 0,
0)], pixel_size: 0.32, magnification: 20', 'created_by': 'microscope',
'file_type': 'ome-tiff'}}
# Task Sequence:
# Image Import: Import the acquired blue and red fluorescence channel images
(OME-TIFF format);
# Image Merging: Merge the imported blue and red fluorescence channel images into
a composite image.

```

The agent loads the individual channel images and combines them into a single multi-channel

composite file, finalizing the execution of the user's task.

```
fiji_initialize()

# Load the blue fluorescence channel image
blue_channel_file = "organoid2_blue.ome.tif"
say("[ACTION] Loading blue fluorescence channel image: " + blue_channel_file)
blue_image = load_image(blue_channel_file)

# Load the red fluorescence channel image
red_channel_file = "organoid2_red.ome.tif"
say("[ACTION] Loading red fluorescence channel image: " + red_channel_file)
red_image = load_image(red_channel_file)

# Merge blue and red channels into a composite image
# Channel order: [red, blue] for merge_channels, colors must match
channels = [red_image, blue_image]
channel_colors = ["Red", "Blue"]
output_file = "merged_organoid2_red_blue.tif"
say("[ACTION] Merging red and blue channels into composite image: " +
output_file)
merged_image = merge_channels(channels, colors=channel_colors,
output=output_file)

fiji_shutdown()
```

This example demonstrates the system's end-to-end operational logic. The workflow begins with the high-level policy's interpretation of a user's scientific goal. Through an interactive refinement loop, this goal is translated into a deterministic, executable plan. The plan is then passed to a low-level policy, where specialized agents orchestrate the control of microscope hardware and the execution of image analysis routines, ultimately achieving the initial experimental objective in a fully automated fashion.

Supplementary Note 4. The Primary Failure Situations from the Model

In deploying a LLM for the autonomous control of complex microscopy and imaging workflows, we identified several distinct categories of failure modes. These errors not only compromise experimental accuracy and efficiency but also elucidate the current limitations of LLMs in comprehending physical instrument constraints, logical operational sequences, and specific domain knowledge in cell biology. Here, we present a systematic categorization and analysis of these observed failures.

1. Hallucination

Hallucination refers to the generation of information or commands that are not grounded in the provided context or factual reality. In the context of microscope control, this manifested as the fabrication of parameters, the introduction of superfluous actions, and deviations from established protocols.

1.1. Fabrication of Arbitrary Parameters

The LLM occasionally generated numerical values for instrument parameters without explicit instruction or a logical basis. For instance, when tasked with performing a Z-stack acquisition of an organoid within the current field of view (FOV), the model erroneously defined a new, arbitrary FOV size (height and width). As the operation was confined to the existing view, stage movement was unnecessary, and redefining the FOV was illogical and could lead to unintended cropping or sampling errors.

User Request:

Imaging Target: 3D Cells;

Imaging Task: Capture the organoids in the field of view along the z-axis and achieve depth-of-field expansion.

Erroneous Command:

```
...
say("Getting recommended Z-stack range for organoid imaging")
z_min, z_max = z_stack_range()
z_min = max(0, z_min)
z_max = min(10000, z_max)
z_step = 7.5
if z_min >= z_max or z_step <= 0:
    raise Exception("Invalid Z-stack range or step")
current_xy = microscope_get_xy_position()
say("Starting Z-stack acquisition at current field of view")
multi_dimensional_acquisition(
    description="Organoid Z-stack imaging, brightfield, 4x objective",
    xy_positions=[current_xy],
    z_range=(z_min, z_max),
    z_step=z_step,
    channels=["1-NONE"],
    exposure_times={"1-NONE": 10},
    height=350,
    weight=350
)
say("Z-stack imaging completed and image sequence saved")
...
```

1.2. Generation of Superfluous Actions

This category includes instances where the LLM appended unnecessary or redundant operations to a given task. A notable example occurred during a brightfield imaging protocol designed to observe cell adhesion and morphology. The model incorrectly introduced a command to acquire a fluorescence signal (“3-U-FBNA”), an action entirely irrelevant to the brightfield imaging task at hand. This not only deviates from the experimental design but could also induce phototoxicity in sensitive live samples.

User Request:

Imaging Target: 2D cells;

Imaging Task: Record every half hour for a total of 12 hours to observe the adhesion and spreading behavior of cells on hydrogels.

Erroneous Command:

```
...
say("Starting 12-hour time-lapse acquisition with 30-minute interval, combining
Z-stack imaging")
current_xy = microscope_get_xy_position()
multi_dimensional_acquisition(
    description="3D hydrogel cell imaging, 20x, Z-stack, 12h, 30min interval",
    xy_positions=[current_xy],
    z_range=(z_min, z_max),
    z_step=z_step,
    channels = ["1-NONE", "3-U-FBNA"]
    exposure_times = {"1-NONE": 10, "3-U-FBNA": 200}
    time_points=24,
    time_interval=1800
)
...
```

1.3. Deviation from Standard Protocols

The LLM sometimes opted for non-standard or suboptimal methods to achieve a goal. For long-term live-cell imaging, maintaining consistent image quality often necessitates adaptive illumination, typically achieved through an auto-exposure routine before each acquisition. Instead of invoking the appropriate auto-exposure function, the LLM hard-coded a fixed exposure value. While technically functional, this approach fails to compensate for dynamic changes in the sample or system, risking image saturation or low signal-to-noise ratios over the duration of the experiment.

User Request:

Imaging Target: 2D cells;

Imaging Task: Capture images every 5 minutes for 12 hours to record the migration process of living cells.

Erroneous Command:

```
...
set_exposure(target_exposure)
say(f"[INFO] Camera exposure time set to {target_exposure} ms to minimize
phototoxicity")
target_brightness = 30
set_brightness(target_brightness)
say(f"[INFO] Light source brightness set to {target_brightness} for long-term
live cell imaging")
say("[INFO] Performing automatic focusing on current field of view containing
living cells")
```

```

optimal_z = perform Autofocus()
set_z_position(optimal_z)
say(f"[INFO] Auto-focus completed, Z position set to optimal value: {optimal_z}
µm")
...

```

2. Misinterpretation

This class of errors arises from the LLM’s inadequate understanding of the physical principles of microscopy, the logical dependencies of operational steps, and the underlying biological context. This results in the incorrect translation of high-level commands into machine-executable instructions.

2.1. Deficient Understanding of Operational Sequences

The LLM frequently failed to grasp the required logical order of operations. For example, in a “find-then-image” task, the protocol required acquiring a low-magnification map to identify organoid locations, followed by high-magnification imaging at those specific locations. The LLM correctly acquired the low-magnification map but then immediately switched to the high-power objective to image the current, unchanged position, entirely omitting the critical intermediate steps of object detection, coordinate calculation, and stage movement.

User Request:

Imaging Target: 3D Cells;

Imaging Task: Use a low-power lens to locate the distribution of cells in the field of view, then use a high-power lens to collect bright field and green fluorescence data in sequence. The operation must be continued for 24 hours, with images taken every 30 minutes.

Erroneous Command:

```

...
say("Switching to low-power 4x objective lens for cell distribution imaging")
microscope_set_objective("1-UPLFLN4XPH")
say("Setting brightfield mode for cell distribution imaging")
microscope_set_dichroic("1-NONE")
say("Setting exposure time to 10ms for brightfield")
microscope_set_exposure(10)
say("Performing auto brightness adjustment for brightfield")
microscope_auto_brightness()
say("Performing auto focus for cell distribution imaging")
microscope_auto_focus()
say("Acquiring brightfield image to locate cell distribution")
cell_distribution_image = microscope_get_current_image("Cell distribution
brightfield image at 4x")
say("Switching to high-power 20x objective lens for detailed imaging")
microscope_set_objective("3-LUCPLFLN20XRC")
say("Configuring brightfield imaging settings at 20x")
...

```

Furthermore, when presented with multiple targets (e.g., *organoid_locations*), the model failed to generate the necessary iterative loop, instead proceeding to image only the first target (*organoid_locations[0]*), thereby neglecting all other identified objects.

User Request:

Imaging Target: Organoids;

Imaging Task: After determining the location of the organoid using a low-power microscope, detailed data collection of the blue and green fluorescence channels at the organoid location is performed using a high-power microscope.

Erroneous Command:

```
...
say("Switching to 40x objective for detailed imaging")
microscope_set_objective("4-LUCPLFLN40X")
say("Loading organoid location from saved file")
organoid_locations = load_target_locations("organoid_locations_list.json")
if not organoid_locations or len(organoid_locations)== 0:
    raise Exception("No organoid locations found in the provided file")
center_x, center_y, width, height = organoid_locations[0]
if not (0 <= center_x <= 50000 and 0 <= center_y <= 50000):
    raise Exception("Organoid location out of stage range")
say(f"Moving stage to organoid location at ({center_x}, {center_y})")
microscope_set_xy_position(center_x, center_y)
say("Configuring filter set and laser for blue fluorescence channel (DAPI)")
...
```

2.2. Conflation of Distinct Imaging Modalities

The LLM demonstrated confusion between different imaging concepts. When instructed to “acquire a green fluorescence Z-stack of the organoid and generate a high-resolution image,” the model correctly performed the Z-stack and applied an Extended Depth of Field (EDOF) algorithm to create a 2D projection. However, it then erroneously concluded the task, mislabeling the EDOF projection as the requested “high-resolution image.” This indicates a failure to distinguish between EDOF, which enhances depth of focus, and techniques such as deconvolution or super-resolution, which enhance spatial resolution.

User Request:

Imaging Target: 3D Cells;

Imaging Task: Green fluorescence imaging of organoids in the field of view along the z-axis and generation of high-resolution images.

Erroneous Command:

```
...
image_path = "x_25000_y_25000_multi_dims"
say(f"[ACTION] Opening Z-stack image file: {image_path}")
```

```

image = open_image(image_path)

say("[ACTION] Performing extended depth of field processing on Z-stack")
edof_image = extended_depth_of_field(image)

output_path = "organoid_edof_result.tif"
say(f"[ACTION] Saving extended depth of field image to: {output_path}")
save_image(edof_image, output_path)
...

```

3. Procedural Errors

These errors reflect the LLM's unfamiliarity with standard operating procedures and best practices in microscopy, leading to suboptimal, redundant, or inefficient workflows.

3.1. Redundant Focusing Operations

Modern confocal microscopes are typically parfocal, meaning that the focal plane is maintained when switching between objectives or fluorescence channels. The LLM appeared unaware of this principle. When tasked with acquiring images in two different fluorescence channels (e.g., blue and green), it redundantly executed an autofocus routine for each channel individually, whereas a single autofocus operation would have sufficed.

```

User Request:
Imaging Target: Organoids;
Imaging Task: Perform blue and green fluorescence imaging of organoids in 96-well
plates every hour.

```

Erroneous Command

```

...
say("Configuring green fluorescence channel: setting filter to 3-U-FBNA and
preparing laser")
microscope_set_dichroic("3-U-FBNA")
microscope_open_laser()
say("Setting exposure time to 200ms and brightness to 0 for green fluorescence")
microscope_set_exposure(200)
microscope_set_brightness(0)
say("Performing auto focus for green fluorescence")
microscope_auto_focus()
say("Configuring red fluorescence channel: setting filter to 4-U-FGNA")
microscope_set_dichroic("4-U-FGNA")
say("Setting exposure time to 200ms and brightness to 0 for red fluorescence")
microscope_set_exposure(200)
microscope_set_brightness(0)
say("Performing auto focus for red fluorescence")
microscope_auto_focus()
say("Preparing for multi-channel, time-series, area scan acquisition")

```

```
x_size = 20000
```

```
...
```

3.2. Inefficient Multi-dimensional Acquisition Strategies

The LLM failed to optimize data acquisition paths. When imaging a 96-well plate for blue and green fluorescence over time, the most efficient method is to acquire all required channels at a single well position before moving the stage to the next well. Instead, the LLM generated two separate, complete multi-dimensional acquisition loops: one to image all 96 wells in the blue channel, followed by a second to image all 96 wells in the green channel. This sequential approach dramatically increases experimental time due to redundant stage movements and introduces a higher risk of spatial misalignment between channels.

User Request:

Imaging Target: Organoids;

Imaging Task: Perform blue and green fluorescence imaging of organoids in 96-well plates every hour.

Erroneous Command

```
...
```

```
say("Starting multi-dimensional acquisition for blue fluorescence in all 96 wells")
```

```
multi_dimensional_acquisition(
```

```
    description="96-well organoid, blue fluorescence, 4x, Z-stack, 24h, 1h interval",
```

```
    xy_positions=well_positions,
```

```
    z_range=(z_min, z_max),
```

```
    z_step=z_step,
```

```
    channels=["2-U-FUNA"],
```

```
    exposure_times={"2-U-FUNA": 200},
```

```
    height=6.5,
```

```
    weight=6.5,
```

```
    time_points=time_points,
```

```
    time_interval=time_interval
```

```
)
```

```
say("Switching to green fluorescence mode")
```

```
microscope_set_dichroic("3-U-FBNA")
```

```
say("Setting exposure time to 200ms for green fluorescence")
```

```
microscope_set_exposure(200)
```

```
say("Setting brightness to 0 for green fluorescence")
```

```
microscope_set_brightness(0)
```

```
say("Turning on laser for green fluorescence")
```

```
microscope_open_laser()
```

```
say("Performing auto focus for green fluorescence")
```

```
microscope_auto_focus()
```

```

say("Starting multi-dimensional acquisition for green fluorescence in all 96
wells")
multi_dimensional_acquisition(
    description="96-well organoid, green fluorescence, 4x, Z-stack, 24h, 1h
interval",
    xy_positions=well_positions,
    z_range=(z_min, z_max),
    z_step=z_step,
    channels=["3-U-FBNA"],
    exposure_times={"3-U-FBNA": 200},
    height=6.5,
    weight=6.5,
    time_points=time_points,
    time_interval=time_interval
)
say("All acquisitions completed and images saved")
...

```

3.3. Omission of Critical Imaging Dimensions

The LLM occasionally neglected essential imaging axes required for specific sample types. When imaging organoids—which are three-dimensional structures—within a large hydrogel drop, the model correctly located each organoid at high magnification but then proceeded to capture only a single 2D image. This demonstrates a critical lack of domain knowledge, as it omits the necessary Z-stack acquisition required to capture the organoid’s full 3D morphology.

User Request:

Imaging Target: Organoids;

Imaging Task: Use a low-power microscope to collect the location of organoids in circular droplets measuring 1 cm in size, then use a 20x microscope to collect the status of the organoids.

Erroneous Command:

```

...
say("Loading organoid positions from detection results")
positions = load_target_locations("organoid_locations_list.json")
if not positions or len(positions) == 0:
    raise Exception("No organoid positions found in detection results")
for idx, (center_x, center_y, width, height) in enumerate(positions):
    if not (0 <= center_x <= 50000 and 0 <= center_y <= 50000):
        say(f"Skipping invalid position: ({center_x}, {center_y})")
        continue
    say(f"Moving to organoid position {idx+1}: ({center_x}, {center_y})")
    microscope_set_xy_position(center_x, center_y)
    say("Performing auto focus at organoid position")
    microscope_auto_focus()

```

```
say(f"Capturing high-resolution image of organoid {idx+1} at 20x")
microscope_get_image_with_size(width, height, f"High-resolution organoid
image {idx+1}, brightfield, 20x")
say("High-resolution imaging of all detected organoids completed")
...
```

4. Incomplete Execution (Instruction Forgetting)

This failure mode is characterized by the LLM successfully executing intermediate steps of a command but failing to complete the entire instruction set, effectively forgetting the final goal. For example, a user requested to deconvolve an image and then display the original and deconvolved images side by side for comparison. The LLM correctly loaded the image, performed the deconvolution, and saved both the original and the processed files. However, it never generated the final command to display the images, thus failing to fulfill the user's primary objective.

User Request:

Imaging Target: 2D Cells;

Imaging Task: Capture the image, perform deconvolution processing on the image, then simultaneously display and compare the effects of the original image and the deconvolved image.

Incomplete command sequence:

```
...
original_output = "original_image_display.tif"
deconv_output = "deconvolved_image_display.tif"
say(f"[ACTION] Saving original image for display: {original_output}")
save_image(original_image, original_output)
say(f"[ACTION] Saving deconvolved image for display: {deconv_output}")
save_image(deconvolved_image, deconv_output)
...
```

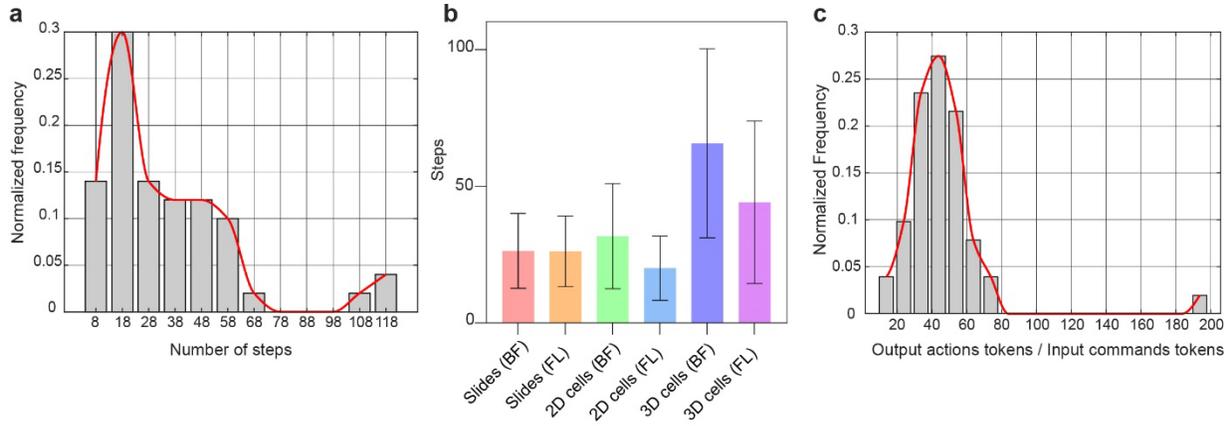


Fig. S1. Step lengths and token consumption of execution actions required for different tasks. **a**, Frequency statistics of instructions with different step lengths; **b**, Distribution of instruction step lengths corresponding to different tasks; **c**, Frequency distribution of the ratio from output action token consumption to corresponding input command token consumption across diverse tasks.

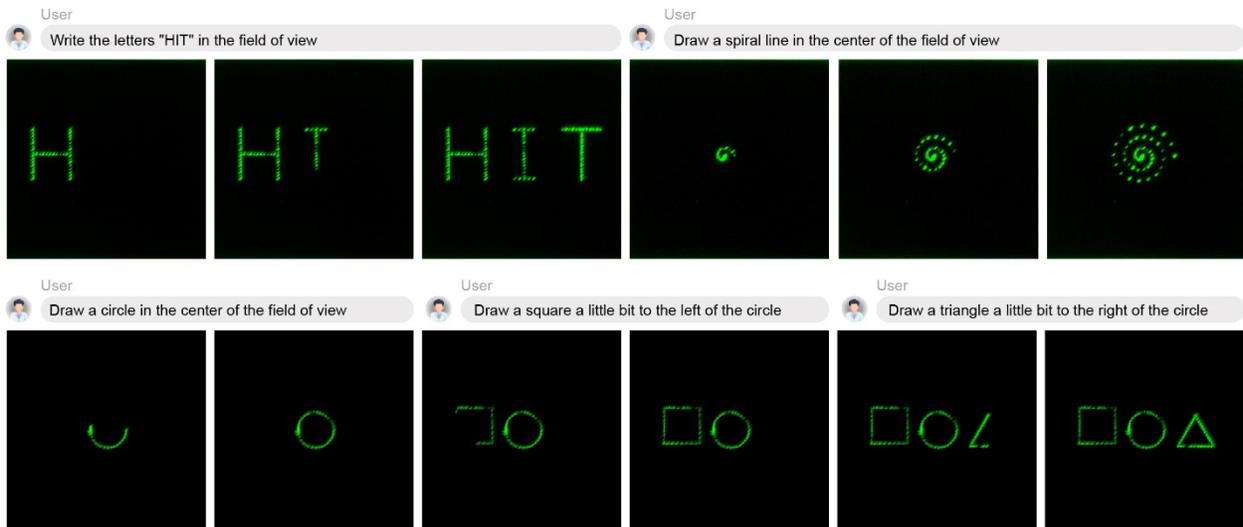


Fig. S2. Pattern generation under spatially resolved microscopy based on FRAP. The first row shows direct commands for drawing letters and spiral lines. The second row presents

sequential commands, where the model infers based on spatial positions to generate distinct geometric shapes in regions with different relative positional relationships.

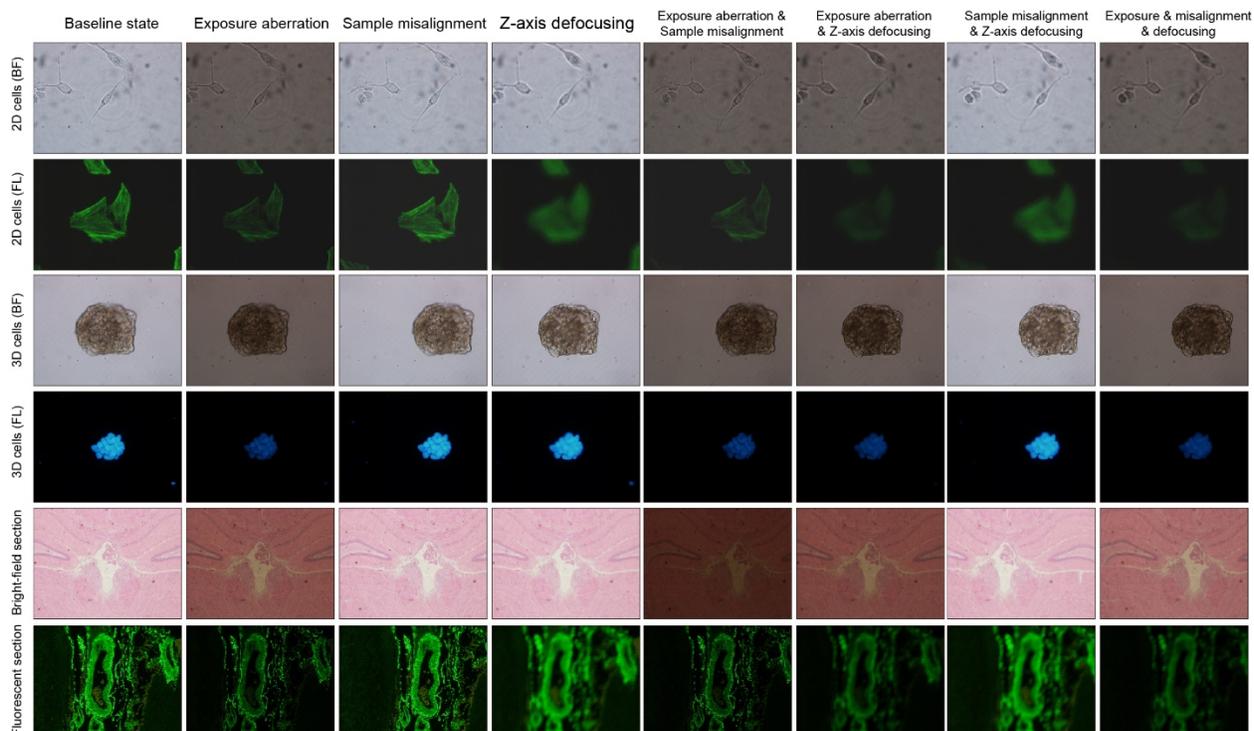


Fig. S3. Imaging characteristics under diverse perturbations across multiple biological specimens. Rows depict different specimen-imaging modality pairs: 2D cells (brightfield (BF) and fluorescence (FL)), 3D cells (BF and FL), brightfield tissue sections, and fluorescent tissue sections. Columns represent distinct perturbation conditions: baseline state, exposure aberration, sample misalignment, Z-axis defocusing, and their combinations. These images demonstrate the system’s performance under various perturbations, aiding in the validation of the closed-loop system’s efficacy.

Table S1. Representative datasets classified based on sample types.

| Sample Type | Commands | Objective Lens | Motorized Stage | Fluorescence Channels | Timer | Z-axis Settings |
|----------------------|--|----------------|-----------------|-----------------------|-------|-----------------|
| Bright-field section | Use the 60x objective to capture the crypt structures in the current field of view. | √ | | | | |
| | Perform a full slide scan using a low magnification objective and return the locations of suspected lesions | | √ | | | |
| | Conduct a 3cm x 3cm area scan with the 4x objective to identify complete organoid locations within the organoid section, then use a high magnification objective for detailed data acquisition at these positions. | √ | √ | | | |
| Fluorescent section | Simultaneously capture multiple fluorescence labels, including DAPI, FITC, TRITC, Cy5, and merge the different channels. | | | √ | | |
| | Capture the blue and green fluorescence channels in the current field of view under both low and high magnification objectives. | √ | | √ | | |
| | Perform a scan and stitch of the entire image under blue fluorescence (nuclear staining), and calculate the density of labeled cells. | | √ | √ | | |

| Sample Type | Commands | Objective Lens | Motorized Stage | Fluorescence Channels | Timer | Z-axis Settings |
|--|--|----------------|-----------------|-----------------------|-------|-----------------|
| | First, use the 4x objective to obtain an overview image of the entire slide, then use the 20x objective for detailed imaging of the blue and green fluorescence channels in suspected tumor regions. | √ | √ | √ | | |
| 2D cells (bright field/fluorescent) | Capture images every 5 minutes for 4 hours to record the morphological changes of cells after drug administration. | | | | √ | |
| | Record every half hour for 24 hours continuously: overall status under low magnification and the stem cell differentiation process in the center of the field of view under 20x magnification. | √ | | | √ | |
| | Use the 4x objective for rapid imaging of cells in a 96-well plate, and quantify cell density and morphological changes every half hour over 6 hours post-drug treatment. | | √ | | | √ |
| | Perform intensive acquisition over 12 hours (e.g., every 10 minutes) to record the movement of mitochondria (labeled with green fluorescence) during cell growth. | | | | √ | √ |
| | Use low magnification to image the entire field and monitor monoclonal formation under sparse plating conditions, then use the 20x objective to image target areas after location to ensure clonal origin purity. Record every hour for 2 days. | √ | √ | | | √ |
| | Use the brightfield channel to record overall morphological changes of multiple cells in the current field of view under 4x magnification after drug addition, and use high magnification (e.g., 40x) to record cytoskeletal changes under the 488-channel. Record every half hour. | √ | | | √ | √ |
| | Use the 4x objective for rapid imaging of cells in a 96-well plate, and quantify cell density and morphological changes every half hour over 6 hours post-drug treatment. | | √ | | √ | √ |
| | Scan a 3x3 area using the 20x objective in brightfield every 30 minutes. Upon detecting cell mitosis, immediately switch to a high magnification objective to capture the process at 1-minute intervals using 560 nm and 640 nm channels. Record for 8 hours in total. | √ | √ | | √ | √ |
| 3D cells (bright field/fluorescent) | Perform Z-stack imaging of the organoid in the field of view and implement extended depth of focus (EDF). | | | | | √ |
| | Record the morphology of the entire gelatin droplet organoid in the current field of view using low magnification (without plate scanning). Then directly switch to the 20x objective to observe the details of one organoid, preferably via Z-stack imaging. | √ | | | | √ |
| | Use the 4x objective to acquire the density distribution of organoids within a 2cm x 2cm gelatin droplet, noting the gel thickness requires Z-stack scanning. | | √ | | | √ |
| | Image the organoid in the field of view by acquiring Z-stacks for green and pink fluorescence, and generate a high-resolution image. | | | | √ | √ |
| | Capture an image every hour for 48 hours with Z-stack scanning to analyze the infiltration and distribution changes of cell spheroids within the hydrogel in the field of view. | | | | √ | √ |
| | Use low magnification to acquire the positions of organoids within a 1cm diameter circular gelatin droplet, then use the 20x objective to sequentially acquire the status of each organoid, requiring Z-stack information. | √ | √ | | | √ |
| | Record the live/dead staining results of a tumor spheroid in a single well of a 96-well plate in the current field of view using the 20x objective. Acquire multiple Z-slices and perform maximum intensity projection. | √ | | | √ | √ |
| | Capture Z-stack images every 15 minutes of the fusion process of organoid assemblies in the field of view: use 4x to record interaction between two microspheres and 20x to record changes at the fusion boundary. | √ | | | | √ |
| | Use the 4x objective for brightfield scanning to obtain a panoramic image of a 3cm x 3cm area, identify and record organoid locations; then switch to the 20x objective to sequentially capture images of each organoid in the blue (DAPI), green (FITC), and red (TRITC) fluorescence channels. | √ | √ | | √ | √ |
| | Continuously for 48 hours, acquire an image hourly to record the aggregation process of cell spheroids in each well of a 2cm square array using low magnification, and simultaneously record one specific well using the 20x objective. | √ | √ | | | √ |
| Continuously for 72 hours, capture an image every 2 hours of the status of organoids in the array to monitor fluorescence signal changes indicative of organoid growth or drug response, using different magnification objectives as required. | √ | | | √ | √ | |
| Switch to low magnification brightfield, acquire a 3x3cm array area, collect brightfield channel information within this area to locate organoids based on brightfield data, switch to the 20x objective, and sequentially acquire the blue and green fluorescence channels for each organoid. This experiment aims to observe organoid interactions and should run continuously for 48 hours, preferably with imaging every 30 minutes. | √ | √ | | √ | √ | |

Note: Nearly all tasks require adjustments for brightness and focus settings, which are therefore not listed separately in the functional description.

Table S2. Representative generalization test dataset.

| Generalization Types | Commands |
|----------------------|---|
| Systematicity | Capture the blue fluorescence channel and apply deconvolution algorithm for processing. |
| | Use the 20x objective for rapid imaging of organoids in a 96-well plate. |
| | Acquire an image of the current state and perform cell segmentation using Cellpose to obtain segmentation masks. |
| | Capture a brightfield image under 20x objective and apply denoising processing. |
| | Under 4x objective brightfield conditions, acquire a 3cm x 3cm image and detect tumor locations. |
| | Under blue fluorescence conditions, use a high magnification 20x objective to capture images of organoids in a 24-well plate every hour for 24 hours. |
| | Turn on the laser and simultaneously capture a red fluorescence image. |
| | At 20x magnification, image organoids in a 96-well plate every hour for 24 hours. |
| | Under 4x objective, acquire a 5cm x 1cm brightfield image and capture the same area using the blue fluorescence channel. Move the Z-axis to the midpoint, perform autofocus, and return the current hardware status. |
| Productivity | Switch to the 4x objective, scan a 5cm x 1cm area, switch to the blue fluorescence channel to scan the same area, detect suspicious areas, move to the first suspicious area, switch to the 20x objective, sequentially activate the blue and green fluorescence channels, and acquire a 3cm x 3cm field of view area. |
| | Switch to the 4x objective, scan a 5cm x 1cm area, detect suspected lesion areas, move to the first detected area, switch to the 20x objective, and acquire a 3cm x 3cm field of view area. |
| | Switch to the 4x objective, scan a 5mm x 1mm area, detect suspected lesion areas, move to the first detected area, switch to the 20x objective, and acquire a 3cm x 3cm field of view area. |
| | Use the 20x objective to capture brightfield, blue, and green fluorescence channel images of a 3cm x 3cm fluorescence section, and perform statistical analysis of nucleus size and count on the blue fluorescence image. |
| | Switch to low magnification brightfield, acquire a 2cm x 2cm array area, locate organoids based on brightfield information, switch to the 20x objective, and sequentially acquire the blue, green, and red fluorescence channels for each organoid. |
| | Use a low magnification objective to locate the distribution of tumor sections within the field of view, then use a high magnification objective to sequentially acquire brightfield and green fluorescence data. This operation must continue for 24 hours, with imaging every 30 minutes. |
| | Use the 4x objective for brightfield scanning to obtain an image of a 3cm x 3cm area, identify and record organoid locations; then switch to the 20x objective to sequentially capture blue (DAPI), green (FITC), and red (TRITC) fluorescence channel images for each organoid. |
| | Switch to the 10x objective, perform a brightfield scan of a 4x4cm area, detect lesion areas, move to the first area, switch to the 40x objective, sequentially activate the green and red fluorescence channels, acquire a 2x2cm field of view area, and measure fluorescence intensity. |
| | Use a low magnification objective (10x) in brightfield mode to scan a 4cm x 4cm area, detect organoid positions, and record their coordinates; switch to the 20x objective to acquire blue and green fluorescence channel images of the first organoid location area. Use the 4x objective to obtain the quantitative distribution of organoids within a 2cm x 2cm gelatin droplet, requiring Z-axis scanning. |
| Substitutivity | First, use the 4x objective to obtain a 5cm x 1cm view and identify lesion locations, then use 20x magnification for high-resolution imaging of a suspected 3cm x 3cm lesion area. |
| | Use the 4x objective to automatically count and locate DAPI-stained nuclei on fluorescence in situ hybridization (FISH) sections. |
| | Simultaneously capture images of multiple fluorescence labels (e.g., TMRM for mitochondria, Hoechst for nuclei, Calcein for cytoplasm) and merge the different channels. |
| | Continuously image cells in a 24-well plate every hour for 24 hours. |
| | Capture the green fluorescence channel image, apply super-resolution algorithms to resolve subcellular structures, and save the image. |
| | Perform blue fluorescence imaging of cells in the field of view and use algorithms to generate a super-resolution image. |
| | Perform Z-stack imaging of the organoid and synthesize into an image. |
| | Capture fluorescence channel images within a 3cm x 3cm area to distinguish live/dead cells (using Calcein-AM staining). |
| | Perform a scan and stitch of the entire image under blue fluorescence conditions, and count the number of labeled cells. Detect weak blue fluorescence signals from low-expression targets. |
| Localism | Acquire the entire petri dish cell status, automatically identify bacterial or fungal contamination in the dish, and issue an alert. |
| | After determining the organoid location using the 4x objective, use the 60x objective for detailed data acquisition at that position. |
| | Perform continuous 24-hour imaging of the green and red fluorescence channels of organoids within a 2cm x 2cm gelatin droplet, acquiring one image per hour. |
| | Use low magnification to acquire the positions of organoids within a 1cm diameter circular gelatin droplet, then use the 20x objective to sequentially acquire the status of each organoid. |
| | Acquire brightfield and different fluorescence signal images, and overlay the fluorescence signals with transmitted light. |

| Generalization Types | Commands |
|----------------------|---|
| | Image organoids in a 96-well plate for blue and green fluorescence every hour. |
| | Capture images every 5 minutes for 12 hours to record live cell migration processes. |
| | Over a continuous 6-hour period, capture images every 30 minutes, acquiring multiple fluorescence labels of the organoids. |
| | Use a low magnification objective to complete a full slide scan, automatically record suspected lesions, and output the locations of all suspicious areas. |
| | Acquire brightfield and different fluorescence signal images, and overlay the fluorescence signals with transmitted light. |
| Overgeneralization | Move the Z-axis to the midpoint of the recommended Z-stack range. |
| | Obtain the current optimal brightness for imaging, then adjust the brightness to half the current level after capture. |
| | Use the 4x objective to acquire a global image, detect bacterial locations, move to the position with the largest bacterial area, and capture an image using the 20x objective. |
| | After capturing an image, detect bacterial locations, and capture images in descending order of bacterial area size. |
| | After focusing on the current target, acquire images at 0.5x, 1x, and 1.5x the current Z-axis position. |
| | After capturing an image, automatically adjust the contrast and use matplotlib (plt) to simultaneously display the processed and original images for 10 seconds. |
| | After capturing an image, perform denoising and deconvolution, then use matplotlib (plt) to simultaneously display the processed and original images. |
| | After acquiring a 3mm x 3mm area image, use the matplotlib (plt) tool to add a scale bar to the image and display it for 5 seconds. |
| | Acquire the current image, detect bacterial areas, annotate the bacterial areas with red boxes using the matplotlib (plt) tool, display for 5 seconds, then close. |
| | Acquire organoid images and obtain the time consumed for image acquisition. |

Table S3. Representative dataset for ambiguous task evaluation.

| Command | Ambiguous types |
|---|--|
| Use a low magnification objective with the DAPI fluorescence channel configured and apply a deconvolution algorithm for deblurring. | Objective lens |
| Capture a single field of view image after focusing with a suitable objective in the brightfield mode and use Cellpose to perform cell segmentation on this image to output a binary segmentation mask. | |
| Use a 20x objective with fluorescence channels configured for continuous imaging over 24 hours; capture an image of each well in a 24-well plate at 1-hour intervals. | Fluorescence channels |
| Use a 4x objective in brightfield mode to identify and record the center coordinates of all organoids by scanning a 3 mm x 3 mm area. Subsequently, switch to a 20x objective, move to the first organoid location, and acquire a multi-channel fluorescence image. | |
| Perform a tile scan and stitch of the entire image under blue fluorescence (nuclear staining), and calculate the density of labeled cells. | Imaging area |
| Use a 4x objective to rapidly image cells in the central wells of a plate, quantifying cell density and morphological changes every 30 minutes over a 6-hour period post-drug treatment. | |
| Perform time-lapse imaging to record the aggregation process of cell spheroids in each well of a 2cm square array using a low magnification objective, while simultaneously recording one specific well with a 20x objective. | Time-lapse acquisition |
| Record the morphological changes of cells after drug administration for a duration of 4 hours. | |
| Perform Z-stack imaging of the organoid in the field of view and use appropriate tools to generate a composite image. | Image post-processing |
| Use a 4x objective to scan a 3cm x 3cm area and process the acquired data using appropriate tools to extract relevant information about the organoids. | |
| Record the position of a tumor spheroid within a single well of a 96-well plate in the current field of view. To obtain fluorescence staining results, switch objectives for acquisition. Ensure to capture multiple Z-slices and perform a maximum intensity projection (MIP). | Objective lens, Fluorescence channels |
| Select a moderate magnification to record the fluorescence staining of the cytoskeleton in the current field of view after drug treatment. | |
| Use a low magnification objective for area scanning to locate intact organoids within the organoid section, then utilize a high magnification objective for detailed data acquisition at these positions. | Objective lens, Imaging area |
| First, use a low magnification objective to image the entire well plate area, monitoring the monoclonal formation process under sparse plating conditions. This entire process lasts for 2 days, with recordings taken every hour. | |
| Continuously over 24 hours, capture the overall state under low magnification and the cell differentiation process in the center of the field of view under a right objective during each imaging cycle. | Objective lens, Time-lapse acquisition |
| Record the fusion process of organoid assemblies in the field of view through time-lapse imaging, acquiring a Z-stack at each time point. | |
| Capture the effects of the blue and green fluorescence channels in the current field of view under both low and high magnification objectives, and process the acquired images. | Objective lens, Image post-processing |
| Use a high magnification objective to capture the ciliary structures of organoids in the current field of view, and select a suitable algorithm to obtain a clear structural image. | |
| Perform a tile scan and stitch of an appropriately sized area in the center of the section under fluorescence conditions, and calculate cell density. | Fluorescence channels, Imaging area |
| Use a 4x objective for brightfield scanning to obtain a panoramic image in the well, identify and record organoid locations; subsequently switch to a 20x objective to sequentially capture fluorescence channel images of each organoid. | |

| | |
|--|---|
| Use the brightfield channel to record overall morphological changes of multiple cells in the current field of view under 4x magnification after drug addition, and use a high magnification objective (e.g., 40x) to record fluorescence-based cytoskeletal changes. The recording duration should be 8 hours. | Fluorescence channels, Time-lapse acquisition |
| Capture images every 5 minutes for an appropriate duration to record the fluorescence signal changes in cells after drug administration. | |
| Simultaneously capture multiple fluorescence labels and process the different channels using appropriate algorithms. | Fluorescence channels, Image post-processing |
| First, use a 4x objective to obtain an overview image of the entire slide, then use a 20x objective for detailed imaging of the fluorescence channels in suspected tumor regions. Simultaneously, apply appropriate algorithm processing to obtain clearer images. | |
| First, use a low magnification objective to image the entire field of view within the current well plate, monitoring the monoclonal formation process under sparse plating conditions. The entire process lasts for an appropriate duration, with recordings required every hour for an appropriate duration. | Imaging area, Time-lapse acquisition |
| Perform intensive acquisition over 12 hours to record the growth process of cells in the entire culture dish, which are labeled with green fluorescence. | |
| Use a 4x objective to complete the scanning of a specific area in the upper left corner of the section, return statistical indicators of the cells in the section, and process the data using algorithms. | Imaging area, Image post-processing |
| Record multiplex immunofluorescence information of cells in the culture dish, primarily using the 488 nm and 640 nm channels, and process these fluorescence images. | |
| Record the morphological changes of cells after drug administration for 4 hours, and process these images for drug efficacy evaluation. | Time-lapse acquisition, Image post-processing |

Movie S1.

Overall introduction and multi-step experiment with fluorescence slice data acquisition and analysis.

Movie S2.

Error correction experiment with fluorescence slice data acquisition and analysis. The test included offset perturbations, defocusing, and overexposure.

Movie S3.

Cross-platform generalization experiments. This includes large language model-driven operation for FRAP experiments.

Movie S4.

Comparison with alternative embodied AI architectures. This includes micromanipulation task performance evaluation with complex dynamics.