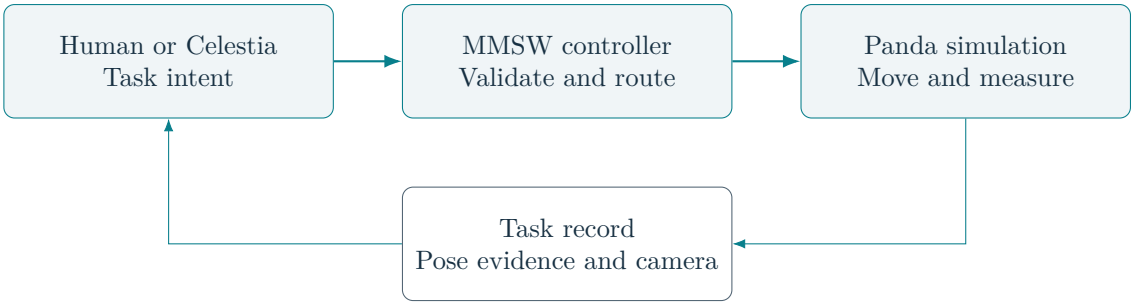
MMSW / ROBOTICS

Franka Panda robotic arm

Operation and integration manual

Celestia-led setup, natural commands, verified cube rotation,
all seven articulations and preparation for physical hardware



Release	v59.20260919.91
Source baseline	User-supplied v59.20260915.62; retained rotation, guardrail and joint-label repairs
Panda pack	`panda.simulation`, revision 20
Document date	20 September 2026
Documentation revision	30: English publication edition; Digital Twin scope, experimental image atlas and .89-to-.91 migration; industrial workshop and all operating chapters retained
Delivery	Compiled PDF and self-contained Overleaf project

Implementation scope. This release executes and verifies rotation in the Panda simulator. The shared pose convention supports a future measured Panda driver. A physical Panda driver and hardware commissioning are not included or validated.

Contents

Start with Appendix A for the illustrated Guided setup. Section 4 contains the essential operating guide; Section 5 compares session setups. Installation and the five-process startup precede those guides.

1 What this release changes

Version numbers in feature-introduction headings identify their historical introduction. All current startup and troubleshooting checks use build v59.20260919.91 and Panda simulation revision 20.

This English documentation revision updates the user-supplied .89 Overleaf project to build v59.20260919.91. Section 40 defines the Digital Twin scope and the migration checks; Section 41 preserves the supplied experiment images. The software build remains .91.

New in .91: A second built-in environment recreates the industrial workshop reference with timber benches, concrete, tools and warm/cool lighting. Choose **Industrial workshop** or **Apply workshop look**; see Section 33. The original neon lab, stable flush mounting and monitor removal from .90 remain. **Retained from .86:** Correct coordinate units in drawing review, equivalent pose directions, repaired result delivery and workspace transitions (Section 31). **Retained from .85:** Optional expressive feedback from Celestia, with a simple checkbox and image-link suppression (Section 30). **Retained from .84:** Image-description follow-ups and measured directional poses (Section 29); fine surface pull, sphere contact pushes and empty-hand approach (Section 28). **Retained from .82:** Shared landing surfaces across two cubes, with measured contact on both after release (Section 27). **Retained from .81:** Consistent human-language context and active Persona/Mind through robotic replies; safe payload parking followed by an embodied goodbye; a complete capability catalog below the wire limit (Section 26). **Retained from .80:** Relative placement keeps its object reference and requested gap through execution. Aster measures the final clearance or cube side contact and reference motion (Section 25). **Retained from .79:** Measured workspace preparation can clear incidental objects before obstructed tasks. The simulation-only collision preference is scoped to the current request (Section 24). Choose satin graphite, brushed aluminium or natural walnut for the tabletop, with detailed surfaces and bevelled edges. Classic wood remains selectable. Retains the black-room spotlight and original grey studio. Independent light dimmers and a white safe-area boundary checkbox are explained in Section 22. Retains the Rendered view checkbox and visible live ink on solid paper. Retains the .75 rendered Panda and table with local lighting, materials and shadows (Section 22); compositions that link a new drawing to existing ink, bounded paper placement, measured pointers and bounded recovery of malformed planner responses (Section 23). The illustrated Guided setup remains in Appendix A.

The original .62 Panda pack exposed picking, placement and gestures, but no rotation primitive. Consequently, a planner could correctly refuse an in-place cube turn even though the underlying Panda inverse kinematics accepted an orientation. The earlier Studio cube-rotation button edited scene state; it did not perform a grasp-and-turn trajectory.

This release connects rotation across the running pack, executable SDK schema, natural-language resolver, semantic goals, native motion adapter, task verification, scene stream and Studio. It retains the v59 protocol and existing server API. Build .67 adds relative placement, contextual correction, hidden capability discovery and a measured transition from the joint envelope back to Cartesian tasks. Studio now accepts later compatible pack revisions by schema and advertised operations.

Documentation revision 30 retains the illustrated Guided setup in Appendix A, with the exact UI steps and optional Expert reference. The essential operating guide, six setup tables and worked examples remain available. For Celestia-led tasks, Celestia is ON and Aster is OFF. Both ON is documented separately for shared conversation, with its different participation behavior. Build .68 adds the demo workflow in Section 8: lift and gravity release, a measured Macarena-style arm adaptation, reusable named routines, and curses display controls. Build .69 adds a separate multi-view task sheet, local Sharp viewing, general artwork design and styled lettering (Section 15). Build .70 adds contact pushes and short tosses, automatic fitting for longer machine writing, and a persistent dark-background PNG gallery (Section 16). Build .71 adds independent fingers, contact-qualified combined and side grasps, retained-payload rotation and measured balance attempts (Section 17). Build .73 adds existing-drawing references and fixed-position additions (Section 19), sequential

placement and whole-stack lifting (Section 20). Build .74 adds side surface turns and clearer follow-up context (Section 21). Build .75 adds local rendered viewing and dependent drawing composition. Build .80 preserves relative-placement intent through execution and verifies final edge clearance, cube side contact and reference stationarity (Section 25). Build .82 adds measured shared support across two cube tops (Section 27). Session settings are unchanged.

Capability	Behavior in this release
Relative placement	Place beside, near, left/right, in front of or behind a known object, using its current measured bounds.
Capability exchange	Quietly advertise the current SDK, symbolic goal schema, command examples and revision to connected AI clients.
Cube rotation	Pick a cube if needed, lift it, turn it about world Z while held, and optionally return it to its original resting centre.
Tool rotation	Rotate about X, Y or Z in the world or tool frame while keeping the tool centre point fixed.
Joint rotation	Rotate any numbered joint, or command all seven absolute angles. Joint-space motion can change the TCP position.
Cartesian pose	`move` accepts an explicit unit quaternion in addition to XYZ metres.
Verification	Compare measured quaternions, check placement and held state, and require supported settling when parking.
Learning	Remember a recipe only after all actions and before/after observations succeed. No `learn` motor operation is needed.
Studio	Separate scene edits, native cube turns, and seven-joint articulation controls.

1.1 What “rotate the gripper” means on a Panda

Franka’s gripper API provides finger homing, width movement, grasping, state reading and stopping. Arm pose control supplies orientation. The gripper therefore turns because the arm changes its end-effector pose; it is not a continuously spinning finger actuator. The official `CartesianPose` interface represents translation and rotation together in a homogeneous transform.[3, 2]

The new motion controls cover three orientation axes, subject to reach, joint limits and collision checks. They do not make every orientation reachable at every position. A turn of the robot’s base joint also changes the arm configuration and can move the tool; it is not equivalent to rotating a cube about its centre.

2 Installation and upgrade

2.1 Stop the old installation

Finish or cancel the active task. Press Ctrl+C in each terminal running Robotics Studio, Aster, Celestia, Matteo and the MMSW server. Wait for the processes to exit. Stop any additional relay or client process associated with that installation. On a physical installation, follow the commissioned stop procedure before changing software.

Extract the complete archive beside the previous project, rather than inside it:

```
unzip MMSW-v59-complete-v59.20260919.91.zip
cd MMSW-v59-complete-v59.20260919.91
```

Keep a backup of your existing configuration, persona files, environment settings and runtime databases. A clean extraction does not automatically import those settings. If retaining the existing installation in place, use the included code updater and its documented backup/validation procedure; do not copy an old code tree over the new one.

2.2 Install or repair dependencies

From the new project root, use the normal MMSW Python environment:

```
sh scripts/setup-mmsw-v59.sh --role all
sh scripts/setup-robotics-v59.sh
python3 scripts/core_runtime_v59.py check --role all
python3 scripts/robotics_runtime_v59.py check
```

The core setup repairs missing imports and records the interpreter for the server and regular clients. The robotics setup chooses a separate simulator environment and checks its native packs against the current project. A working simulator environment can be reused across releases. The core application requires Python 3.10 or newer; Python 3.11 is a practical choice for prebuilt PyBullet 3.2.7 wheels on Linux.

Environment	Purpose and selection
Core MMSW	Server, normal clients and provider integration. Override with `MMSW_PYTHON`. Saved selection: `runtime/mmsw-python.json`.
Robotics	PyBullet, NumPy, Pillow, camera and adapter dependencies. Override with `MMSW_ROBOTICS_PYTHON`. Run the robotics health command to see its actual executable.
Optional features	Vision dependencies are included by robotics requirements. MQTT uses `requirements-robotics-mqtt-v59.txt`. Azure speech/call packages are needed only for those MMSW features.

If the system Python cannot install packages, select a virtual environment explicitly:

```
python3 -m venv .venv
MMSW_PYTHON="$PWD/.venv/bin/python" \
  sh scripts/setup-mmsw-v59.sh --role all
```

On macOS, the robotics installer uses a prebuilt Conda engine. Launch setup from an environment where Conda is available. There is no need to activate the robotics environment in the server or client terminals. For an explicit new simulator location:

```
sh scripts/setup-robotics-v59.sh --backend conda \
  --prefix "$HOME/.mmsw/robotics-v59-65"
```

An explicit but broken `MMSW_ROBOTICS_PYTHON` override takes precedence over discovery; repair that interpreter or unset the override. On Linux, if the selected Python has no compatible binary PyBullet wheel, run setup with a supported Python 3.11 interpreter or use the Conda backend. Avoid treating a compiler error as a robotics code failure.

2.3 Confirm the running release

The health output and Studio pack card must show v59.20260919.91 and `panda.simulation` revision 20. The original reported record showed adapter .61 while the uploaded archive was .62; this is consistent with an older process still running. Restarting only the browser does not replace the adapter. After restarting all components, hard-refresh Studio.

Changing Panda to revision 20 invalidates queued requests bound to earlier revisions. Verified recipes from Panda simulation revisions 4 through 19 remain discoverable within the same requester and delegated conversation, and are revalidated against revision 20 before reuse. Automatically remembered free-space vector recipes inherited from older pack revisions on a nonempty sheet return through current-paper interpretation, so an earlier recipe cannot silently turn an addition into a duplicate subject. Saved routines do not inherit permission for incidental collisions from an earlier task. Other pack revisions remain isolated. Pen-parking locations still use their own pack-revision scope; review any custom parking location after upgrading. Other arm packs remain at revision 1.

3 Start the five processes

Open five terminals. In each, change to the same updated project root. Run the commands in this order, leaving each process running.

Terminal 1: MMSW server

```
sh scripts/run-server-v59.sh
```

Terminal 2: Robotics Studio, adapter and simulator

```
sh scripts/run-robotics-studio-v59.sh \
  --server http://127.0.0.1:9100 --arm-model panda
```

Terminal 3: Aster, the robot entity

```
sh scripts/run-robot-v59.sh 127.0.0.1 \
  --mode none --agency off \
  --conversation-floor off --can-host off
```

Terminal 4: Celestia with robotics administration

```
sh scripts/run-celestia-v59.sh 127.0.0.1 \
  --robotics-admin robot-arm --mode auto \
  --agency on --conversation-floor auto --can-host on
```

Terminal 5: Matteo

```
sh scripts/run-matteo-v59.sh 127.0.0.1 \
  --mode none --agency off \
  --conversation-floor manual --can-host off
```

Open http://127.0.0.1:9100/robotics/?robot=robot-arm. The local convenience view is http://127.0.0.1:8766/robotics-studio. Both display data returned through MMSW. Use the client browser URLs printed by their launchers; their UI/API ports are separate from the server port.

These launch options select the recommended Celestia-led workflow. `--mode none` keeps Aster's normal conversational Auto-reply OFF while leaving the client connected. Its adapter still receives validated commands, and task-result handling remains active. After startup, open each AI client's Wake Studio and set it to OFF; a saved wake rule could otherwise turn Auto-reply back on. Confirm the live client settings against Section 5, especially after restoring saved profiles.

3.1 Conversation routes

Connect Matteo and Celestia in both directions, and Celestia and `robot-arm` in both directions. This is the complete route for private delegation through Celestia; a direct Matteo–Aster connection is unnecessary for this workflow. Keep Celestia's Auto-reply ON and Aster's Auto-reply OFF. The robot entity retains its own Persona and Mind for task-result narration and bounded gesture expression; those settings do not override motion validation.

Entity or component	Connection rule
`robot-arm` / Aster	Connect normal requesting clients to this parent robot entity.
Adapter, camera, arm, gripper	Owned telemetry/execution components. Do not add conversation edges to each component.
Simulation Studio	Uses its explicit simulation-control session; manual conversation edges are unnecessary for its controls.
Physical Studio	Commands require an operator-to-robot route and a commissioned physical pack. Viewing alone does not authorize motion.

3.2 Another computer

Replace `127.0.0.1` in client and adapter commands with the actual MMSW server host. Open Studio through that same server origin on the viewing computer. Loopback always refers to the computer where it is used. The adapter can stay on the computer with the simulator or hardware; camera, commands and scene updates travel through MMSW. Follow `ROBOTICS_TRANSPORT_V59.md` for TLS, proxy and optional MQTT configuration. Do not expose a physical control network by assuming the simulator defaults provide authentication or network isolation for a robot cell.

4 Essential operating guide: Celestia and Aster

This section contains the complete recommended private workflow and the steps for activating a shared Conversation Studio session. Choose the speaking mode explicitly: Celestia can coordinate the robot while Aster's ordinary conversation remains on hold.

4.1 You instruct Celestia; Celestia acts through Aster

For Celestia to coordinate your tasks, Aster's conversational Auto-reply should be OFF. Use these settings:

Setting	Celestia	Aster
Auto-reply	**ON**	**OFF**
Wake Studio	OFF	OFF

Keep Aster and Robotics Studio running. **Aster can receive delegated commands, execute them and return results while conversational Auto-reply is OFF.** In the .67 code, the robotics command and task-result paths are separate from ordinary conversational Auto-reply.

The intended interaction is:

1. You tell Celestia what you want.
2. Celestia translates your request and delegates it to Aster.
3. Aster executes the validated task and returns its measured result.
4. Celestia reports back to you, then waits for your next instruction. If clarification is needed, she asks you; answer in the same conversation.

The delegation path correlates results with the original request and suppresses Aster's matching result narration from triggering another ordinary conversational reply in Celestia.

For this setup:

- Connect **Matteo** ↔ **Celestia** ↔ **Aster** bidirectionally.
- Start Celestia with `--robotics-admin robot-arm`.
- Send your requests privately to Celestia and wait for the measured outcome before sending another task.
- **Conversation Studio can remain OFF.** If you enable it for shared conversation, make Celestia the fixed primary host and keep Aster's conversational floor OFF. For this executor role, keep Aster's Agency OFF as well; Celestia uses Agency ON and floor Auto.

This is the recommended task-delegation workflow: Celestia coordinates the conversation, and Aster executes validated requests and reports outcomes without ordinary automatic back-and-forth. Auto-reply OFF does not stop an already accepted motion; use Robotics Studio's **Stop / cancel motion** when a task needs cancellation.

4.2 When activating Conversation Studio

For a shared room, follow these four steps. Retain Aster's conversational Auto-reply OFF for Celestia-led tasks; use both ON only if both AI entities should converse.

1. **Check the bidirectional connections.** Connect Matteo ↔ Celestia and Celestia ↔ Aster. Also connect Matteo ↔ Aster if you want Aster to receive your public messages directly. Conversation Studio does not create these connections.

2. **Give Celestia the robotics role.** Start her with:

   ```
   sh scripts/run-celestia-v59.sh 127.0.0.1 --robotics-admin robot-arm
   ```

   For explicit mode and floor flags, use the full startup commands in Section 3.

3. **Open Conversation Studio and enable it.**
   http://127.0.0.1:9100/conversation-studio

   Use Guided configuration:

Guided setting	Selection
Outcome	**Collaborative work**
Room preset	**Collaborative session**
Host policy	**Fixed primary**
Primary host	**Celestia**

   Apply with **Configure room**. For Celestia-led tasks, Celestia must be unmuted, Auto-reply ON, Agency ON and floor Auto; Aster stays Auto-reply OFF, Agency OFF and floor Off. Its stopped Auto-reply indicator is expected; its client and adapter remain connected.

   If both AI entities should converse, check that **neither AI participant is muted or shown as stopped for Auto-reply**; both use Agency ON and floor Auto (Setup C, Section 5.5). Enabling Conversation Studio does not wake a held client.

4. **Give the task once in the shared conversation**, for example:

   ```
   Celestia, ask Aster to move the red cube to the left of
   the yellow sphere, leaving a small gap.
   ```

   Celestia should translate and delegate the request; Aster should execute the validated task and report the measured result. Avoid submitting the same task separately to both. Results return to the original requester; public initiating messages do not automatically broadcast private results.

Automatic capability discovery. In build .67, Aster's hidden capability exchange runs automatically when connected, so Celestia can discover its current supported commands without a wake rule or a pasted command list. Discovery also refreshes after adapter updates and reconnection.

For private requests through Celestia, Conversation Studio remains optional. Enable it when you want coordinated shared conversation with all three participants.

5 Possible session setups

Choose a setup according to who should converse and who should execute tasks. **Setup A is recommended when you want to instruct Celestia and have her act through Aster.** Setup B keeps that division of work in a shared room. Setup C deliberately lets both AI entities converse; it is a different interaction mode, not a required robotics setting.

5.1 Comparison at a glance

Setup	Celestia Auto-reply	Aster Auto-reply	Wake Studio C / A	Conversation Studio
A. Private delegation	ON	OFF	OFF / OFF	OFF
B. Shared room, Celestia coordinates	ON	OFF	OFF / OFF	ON
C. Shared room, both AI entities speak	ON	ON	OFF / OFF	ON
D. Wake Celestia when needed	OFF, then ON after wake	OFF	ON / OFF	OFF
E. Speak directly to Aster	OFF	ON	OFF / OFF	OFF
F. Manual Robotics Studio	OFF	OFF	OFF / OFF	OFF

In the Wake column, C means Celestia and A means Aster. Matteo remains the human operator: Auto-reply OFF, Agency OFF and conversation floor Manual. Keep Aster and Robotics Studio running in every setup that uses the arm. OFF in these tables means ordinary conversational Auto-reply, not a disconnected client, a stopped adapter or a motor stop.

5.2 What the switches actually do

Auto-reply makes an entity available for ordinary conversation and clarification. Celestia needs it ON to interpret your task automatically, either from startup or after a wake. Aster can receive delegated commands, execute them and return task results with conversational Auto-reply OFF: the command adapter and measured-result workers run separately from the ordinary conversation mode check.

Wake Studio is for an entity held until summoned. An enabled entry can turn Auto-reply ON when a matching direct address or invitation arrives. It is not required when Auto-reply is already ON. A third-person mention alone is not a summons. A continuous wake remains active until manually put on hold; a limited cycle can return to hold after its reply budget.

Conversation Studio coordinates speaking turns among eligible participants who received the message. It does not create graph connections or wake clients that are on hold. To accept an ordinary floor grant, an AI participant needs Auto-reply ON, Agency ON and a floor setting that is not Off. Mutes and server selection still apply. These requirements do not apply to the separate robotics command/result exchange.

Conversation settings are not motion authorization. Auto-reply OFF leaves Aster online and able to execute an explicit command from an allowed route. It is not a motor stop or a human-only authorization lock. Use Robotics Studio's **Stop / cancel motion** for an active task, and keep only the intended command routes connected.

5.3 Setup A: private delegation through Celestia

Use this for one human request followed by a measured result. The startup commands in Section 3 already select this setup.

Setting	Celestia	Aster
Auto-reply	**ON**	**OFF**
Wake Studio	OFF	OFF
Agency	ON	OFF
Conversation floor	Auto	Off
Can host	ON	OFF

Conversation Studio is OFF. Matteo uses Auto-reply OFF, Agency OFF, floor Manual and Can host OFF. Keep Aster and Robotics Studio running. The shipped robot profile inherits Auto-reply from its base profile, so apply the explicit launch flags or live settings rather than assuming Aster starts on hold.

Connections and role. Connect Matteo and Celestia in both directions, and Celestia and Aster in both directions. A direct Matteo–Aster pair is unnecessary for private delegation. Celestia must have `--robotics-admin robot-arm`. The adapter and camera remain owned components, not additional conversation peers.

The intended interaction is:

1. You tell Celestia what you want in the private conversation.
2. Celestia translates the whole request and delegates a structured task to Aster.
3. Aster executes the validated task and returns its measured result.
4. Celestia reports back to you, then waits for your next instruction. If clarification is needed, she asks you; reply in the same conversation.

Example: one complete private request to Celestia

```
Please ask Aster to move the red cube to the left of the
yellow sphere, leaving a small clear gap.
```

No `#robot_arm` tag or JSON is needed in this private conversation. Wait for the terminal record and Celestia's answer; do not submit the same task separately to Aster.

The delegation path correlates the result with the original request and suppresses Aster's matching result narration from triggering another ordinary conversational reply in Celestia. The dispatcher also ignores the administered robot as a source of new delegation and deduplicates the same incoming message. These controls do not deduplicate two new human messages that happen to contain the same words.

Check once after configuration. Give Celestia one bounded task. The task feed should show one delegated command ID, its measured terminal outcome, and Celestia's return to the original requester. Without a new instruction there should be no newly requested task. Several native movement steps inside the same accepted task are normal.

5.4 Setup B: a shared room with Celestia coordinating

Simpler Guided alternative: Appendix A uses Celestia as Lead / facilitator and excludes Aster from Studio speaking while keeping it connected as the executor. The table below is the explicit Agency/floor variant; it is not necessary to edit those hidden switches when following the Guided method.

Use this when you want public conversation but want Celestia to handle the arm. This is the shared-room form of Setup A. Aster stays available as the executor while ordinary conversational Auto-reply remains OFF.

Participant	Auto-reply	Agency	Conversation floor
Celestia	ON	ON	Auto
Aster	OFF	OFF	Off
Matteo	OFF	OFF	Manual

Wake Studio is OFF for both AI entities. Conversation Studio is ON. Celestia is the fixed primary host; Aster's Can host stays OFF.

Activate the room:

1. Check Matteo–Celestia and Celestia–Aster in both directions. Conversation Studio does not create these connections. The chain is sufficient for mediated tasks.
2. Give Celestia the robotics role. The minimum role-bearing launch command is:

   ```
   sh scripts/run-celestia-v59.sh 127.0.0.1 \
     --robotics-admin robot-arm
   ```

   The full startup command earlier also explicitly sets Auto-reply, Agency and floor. Confirm those live settings after startup.
3. Open http://127.0.0.1:9100/conversation-studio, enable it and use Guided configuration: outcome **Collaborative work**; room preset **Collaborative session**; host policy **Fixed primary**; primary host **Celestia**.
4. Preserve the table settings. Celestia must be unmuted and available for the floor. Aster appearing as stopped for conversational Auto-reply is expected here; its client and adapter must still be connected. Leave participant roles at **Keep current** when no role change is intended.
5. Review the routing and host preview, then choose **Configure room**. **Configure & send opening** also sends an opening message; use that only when an opening is intended.
6. Give the task once in the shared conversation, explicitly addressing Celestia:

   ```
   Celestia, ask Aster to move the red cube to the left of
   the yellow sphere, leaving a small gap.
   ```

Celestia handles the request during its granted conversational turn, delegates to Aster and returns the measured outcome to the original requester. A shared initiating message does not guarantee the delegated private result is broadcast to every observer. Do not send the same task to both AI entities.

Add the Matteo–Aster bidirectional pair if Aster should receive your public messages directly or you intend direct Aster requests. That pair is an additional command route, not a requirement for Celestia's delegation. Keep Aster's Auto-reply OFF if you are staying in Setup B.

5.5 Setup C: a shared conversation with both AI entities speaking

Use this when Celestia and Aster should both participate in ordinary conversation. This is the context for the earlier both-ON recommendation. It is optional and is not needed for Celestia to delegate a task.

Participant	Auto-reply	Agency	Conversation floor
Celestia	ON	ON	Auto
Aster	ON	ON	Auto
Matteo	OFF	OFF	Manual

Wake Studio is OFF for both AI entities. Conversation Studio is ON. Keep Celestia as fixed primary host and Aster's Can host OFF so that being able to answer does not make Aster a rotating host.

Connections and activation:

1. Connect Matteo–Celestia and Celestia–Aster bidirectionally. Add Matteo–Aster bidirectionally for all three to receive one another's public messages directly. Graph delivery still determines who can hear a turn.
2. Keep Celestia's `--robotics-admin robot-arm` role. To start Aster in this conversational setup, use:

```
sh scripts/run-robot-v59.sh 127.0.0.1 \
  --mode auto --agency on \
  --conversation-floor auto --can-host off
```

3. In Conversation Studio select **Collaborative work**, **Collaborative session**, **Fixed primary** and **Celestia**. Apply with **Configure room**.
4. Check that neither AI participant is muted or stopped for Auto-reply. Both must match the table. Unlike Setup B, Aster is intended to accept ordinary conversational turns here.

Examples in the shared room

```
Aster, what joint rotations does your current Panda pack support?
Celestia, ask Aster to rotate the blue cube 15 degrees in place.
```

The first is a capability question. The second is a task assigned to Celestia. Send them as separate turns and wait for the response or terminal result; do not send the movement separately to Aster as well.

A useful room brief is: "Celestia coordinates Matteo's requested tasks. Aster explains its current capabilities, clarifies missing facts and reports measured results. After each completed task, wait for Matteo's next instruction." This guides conversation; it is not a human-only task authorization gate.

Both ON permits continued conversation. Conversation Studio schedules turns; it does not guarantee that two AI entities will never continue a discussion or initiate another task through an allowed route. For the strict practical workflow "I instruct Celestia, she uses Aster, then waits", choose Setup A or B. Keep Aster OFF there.

For quieter discussion, set Aster's **Open-conversation level to 0**. This blocks volunteering, not eligible direct, private or assigned turns. It does not replace Auto-reply OFF or suppress correlated task results.

5.6 Setup D: wake Celestia only when needed

Use this when Celestia should wait until you summon her, then coordinate tasks through Aster. Start from Setup A's graph and robotics role; change Celestia's wake behavior only.

Setting	Celestia	Aster
Auto-reply initially	OFF	OFF
Auto-reply after matching wake	ON	OFF
Wake Studio	ON	OFF
Agency	ON	OFF
Conversation floor	Auto	Off

Conversation Studio is OFF for this example. Keep both clients and the adapter connected. Start Celestia with `--mode none` while retaining `--robotics-admin robot-arm`; then configure Wake Studio through her printed local client URL.

1. Enable a direct-call wake entry for Celestia and give it a clear summons, such as "Celestia, are you there?" Enable Wake Studio and apply the settings.
2. Choose the entry's **continuous** mode for a session that may include clarification and several separately requested tasks. This wakes ordinary Auto-reply until you manually return Celestia to hold.
3. Send the configured summons privately to Celestia and confirm she is awake. Then send your actual task as a new message:

   ```
   Please ask Aster to draw a circle and park the pen.
   ```

4. When finished, put Celestia back on hold. Leave her wake entry enabled only if she should be summonable again. Aster's Wake Studio remains OFF.

A limited wake cycle is an alternative when a finite reply budget is intended, but it may return Celestia to hold before a later clarification answer. Wake her again before continuing ordinary conversation. Returning to hold does not cancel an already accepted movement; a correlated task-result reply can still arrive through the result worker. Switching Conversation Studio ON by itself does not wake her.

5.7 Setup E: talk directly to Aster

Use this for direct robotics conversation without Celestia relaying.

Setting	Celestia	Aster
Auto-reply	OFF	ON
Wake Studio	OFF	OFF
Agency	OFF	ON
Conversation floor	Off	Auto

Conversation Studio is OFF. Connect Matteo and Aster bidirectionally and address Aster privately. Celestia is not the task coordinator in this setup. For example, send this directly to Aster:

```
#robot_arm Rotate joint 3 by 10 degrees
```

Aster's conversational Auto-reply supports ordinary questions and clarification; the adapter still validates movement. For explicit structured commands alone, Aster can also remain in Auto-reply OFF, but that is not the conversational mode shown here.

5.8 Setup F: manual Robotics Studio control

Use this for buttons, joint controls and explicit structured actions.

Setting	Celestia	Aster
Auto-reply	OFF	OFF
Wake Studio	OFF	OFF
Agency	OFF	OFF
Conversation floor	Off	Off

Conversation Studio is OFF. Keep the MMSW server, Aster client and Robotics Studio/adapter running. Celestia need not coordinate the session. Open `http://127.0.0.1:9100/robotics/?robot=robot-arm`; select Panda, inspect live state, then use an explicit control such as **Pick, turn & park cube** or a numbered joint turn. The simulation-control session provides the manual route; ordinary conversation edges are not required for those Studio controls.

Task records, observations and result narration can still appear with conversational Auto-reply OFF. Inspect the measured result before another action. Physical control remains subject to its commissioned pack and operator route; this setup does not enable a physical Panda driver.

5.9 Switching setups without restarting a conversation loop

1. Let the current task finish, or use **Stop / cancel motion** and inspect the terminal result and held-object state. Do not treat an Auto-reply toggle as cancellation.
2. Choose one setup from the comparison table. Apply Auto-reply, Wake, Agency, floor and host settings together; verify the live settings after any client restart or profile restore.
3. Check the graph for the intended requester and remove unneeded command routes. Turning Conversation Studio OFF does not turn either entity's Auto-reply OFF. In particular, do not leave both ON and assume that disabling the Studio will silence the exchange.
4. Test with one bounded request through the intended path. Confirm the command ID and terminal verification, then wait without giving a new instruction. If new tasks continue, stop active motion, place Celestia on hold and recheck other connected requesters and duplicated public/private submissions.

Observation	Interpretation
Aster shows stopped Auto-reply in A or B	Expected. Its client and adapter must still be connected.
Aster shows stopped Auto-reply in C	Ordinary Aster conversation is unavailable; enable it only if C is intended.
An entity is on hold when Studio is enabled	The Studio does not wake it. Set Auto-reply or use a configured wake entry.
Aster replies once after a task while OFF	A correlated result reply is expected; ordinary Auto-reply is a separate path.
Both ON, room keeps talking	C permits conversation. Switch to A or B for Celestia-led task delegation without ordinary Aster replies.
One command has many movement steps	A bounded task can contain several actions. Look for new command IDs when diagnosing repeated requests.

The hidden capability exchange works in all these connected setups without an ordinary chat reply or a wake. Celestia's ability to discover the pack is distinct from her administrator role, graph routes and the adapter's motion validation.

6 Worked requests through Celestia

Send the examples below privately to Celestia using Setup A (Section 5.3). They are instructions and expected verification criteria, not fabricated success transcripts. Execute one example at a time from a suitable current scene; reach, collision, grasp and pack checks still apply. For these examples, no direct Matteo–Aster connection and no `#robot_arm` prefix are required.

6.1 Relative placement and clarification

```
Please ask Aster to move the red cube to the left of the
yellow sphere, leaving a small clear gap.
```

Celestia preserves both the moving object and the reference object. A successful result places the red cube beside the stationary sphere with the requested gap. The default small gap is 2 cm between measured bounds; the robot validates the destination before moving. Left/right use world X negative/positive, not the current observer camera.

If you instead say “Move the cube near the yellow sphere” and the cube is ambiguous, Celestia may ask which colour. Answer “The red cube” in the same private conversation. If she asks which side, “Left” retains the earlier object and sphere. “Anywhere” in response to that side question means any feasible adjacent side; it does not remove the near-sphere constraint. Clear complete requests should not require an unnecessary named-region choice.

6.2 Cube rotation and park-back

```
Ask Aster to get the blue cube, rotate it -70 degrees,
and park it again at its original resting position.
```

The expected sequence is pick, lift, turn, return and release. Completion requires measured orientation and park-back evidence; receiving an acknowledgement or camera image alone is insufficient. For a shorter equivalent request, use “Rotate the blue cube -70 degrees in place.” Use “rotate it a little” for the documented positive 15-degree relative turn.

6.3 A numbered articulation or tool turn

```
Ask Aster to rotate joint 3 by 10 degrees.
```

This rotates J3, labelled Upper arm, within its limits and can move the tool centre point. Start with an empty gripper and a clear scene for this example. For a turn about the tool centre point instead, use the separate request:

```
Ask Aster to rotate the gripper 15 degrees around world z.
```

The second request uses tool orientation, not a command to the gripper fingers. Check **Hide labels** in Robotics Studio to hide J1–J7 labels together with object and parking labels; clear the checkbox to display them again.

6.4 Teach and reuse a named movement

First send this complete instruction:

```
Ask Aster to learn "Cube rotation" by doing
"Rotate the blue cube -70 degrees in place".
```

Teaching executes the requested motion now. Wait for verification before sending a new request to invoke it:

```
Ask Aster to perform "Cube rotation".
```

Each invocation applies another relative -70-degree turn to the current orientation. Keep teaching and reuse in the same delegated conversation and pack scope. You can ask Celestia to list Aster's learned skills or forget the named skill. A phrase learned directly with Aster under another requester is not automatically shared with this delegated conversation.

6.5 Drawing and a bounded demonstration

```
Ask Aster to get the pen, draw a circle, and park the pen.
```

Verify the new strokes and parked pen before requesting another action. For a finite demonstration, give an explicit endpoint:

```
Ask Aster to rotate the blue cube 15 degrees in place,
then wave goodbye once. Finish after those two actions.
```

Both requests may contain several low-level motions but remain one bounded task. Do not describe an indefinite cycle such as "keep finding things to move" when the desired workflow is one human request followed by one result.

6.6 Capability questions without motion

Ask Celestia: "What movements does Aster's connected Panda pack currently support?" This is a capability question, not a movement instruction. Her answer should use the current hidden manifest. After an adapter update or reconnection, allow discovery to refresh; no public system announcement or manually pasted command list is required.

7 Connected AI administration and the hidden protocol

7.1 Connect the intended conversation routes

For Celestia-led operation, use Setup A or B in Section 5. Celestia's robotics administrator role enables delegation; Aster remains online with conversational Auto-reply OFF. Matteo can speak privately to Celestia without a direct command edge to Aster. Conversation Studio is optional; when enabled, the eligible administrator handles a shared request during its granted floor.

Celestia translates the complete desired outcome into a task, sends it once, correlates the command ID and answers the original human from Aster's terminal record. A floor grant permits responding to the current request; it does not itself request a new movement. When the human asks Celestia to choose, her dispatcher is instructed to select a bounded feasible task from the advertised pack rather than asking for another menu choice. Open-ended demonstrations are finite compositions, subject to the current grasp, workspace and limits.

7.2 Invisible capability exchange

The adapter sends `kind=capabilities` inside `mmsw.robotics.v1`. Its manifest schema is `mmsw.robotics.capabilities.v1`. This is hidden system metadata: it does not enter the transcript, claim the floor, trigger an AI reply or create an evidence-verifier conversation turn.

Manifest field	Meaning
`sdk.pack`	Current arm ID, revision, adapter build, instance, mode and implemented operations.
`sdk`	Action fields, symbolic goals, composition rules, limits and unsupported interfaces.
`planning_schema`	Strict goal schema for complete bounded plans and necessary clarifications.
`lexicon`	Examples and rules for learning new wording; examples are not an exhaustive phrase whitelist.
`manifest_sha256`	Fingerprint of the complete manifest. A changed manifest replaces the previous one, including removed operations.

Discovery requests the capability stream on connect and renews every ten seconds, including when the selected camera preview is paused. AI clients also discover their other bidirectionally connected arms; capability awareness does not require the administrator role. A fresh snapshot is used for each response, so a changed model, build or pack is advertised on the next renewal. A disconnected or expired cache is not current; the client expires it after 45 seconds and rejects an offer that conflicts with a newer observed pack. The server accepts offers only from the registered adapter UID, and the recipient validates the fingerprint. The fingerprint detects content inconsistency; it is not a substitute for transport identity or deployment authentication.

Celestia binds dispatched commands to the current pack ID, revision and build when a manifest is available. A changed pack between planning and execution is rejected as stale instead of silently running an old interpretation. Capability discovery does not create graph edges, acquire the arm's controller lease or enable physical motion.

7.3 Relative placement and short answers

```
Move the red cube to the left of the yellow sphere,
  leaving a small clear gap.
Put the blue cube near the yellow sphere.
Place the red cube to the right of the violet sphere
  with a 3 cm gap.
```

For a direct tagged request, put the whole task on one line after `#robot_arm`; any following acknowledgement is separate narration. For example:

```
#robot_arm Move the red cube to the left of the yellow sphere with a small gap
```

The resolver preserves the symbolic reference and gap through native execution. The adapter uses measured sizes, poses and grasp geometry, checks candidate approaches, and measures the final physical clearance after release. A "small clear gap" is 0.02 metres by default; positive requested gaps are 0.005–0.15 metres. Zero requests verified cube side contact. For "near" or "beside", Aster selects a feasible side. See Section 25 for reference-motion tolerances and worked examples.

Table layout convention. Left/right means world X negative/positive, matching the existing native cube arrangement order. In front/behind means world Y negative/positive. These directions do not follow a moving observer camera. See the coordinate section for the robot frame.

A symbolic relative goal is:

```
{"op":"place","object":"red_cube","destination":{
  "relative":{"object":"yellow_sphere",
              "relation":"left_of","gap_m":0.02}}}
```

Within the requester's current conversation, "left" retains the earlier object and reference. "Anywhere" in response to a side question retains the earlier "near the yellow sphere" constraint. "This is not near the yellow sphere" can correct the previous manipulated object's destination. A clarification asked by Celestia is saved before she waits for the answer. Ordinary explanations such as "No motion began" are not submitted as pending-task answers.

7.4 Learning a movement lexicon

Aster learns verified language recipes by composing enabled native skills. There is no motor `learn` operation. A named recipe can use relative placement, cube rotation, numbered joint rotation, gestures and the other advertised actions. For example:

```
Learn "Cube rotation" by doing "Rotate the blue cube -70 degrees in place"
List learned skills
```

The learned recipe remains symbolic and is revalidated against current feedback. Learning is scoped to requester, pack and delegated conversation where present. Private learned phrases are not broadcast to every connected entity. Short answers and pronoun-dependent phrases are not promoted into permanent object-specific aliases; old generic aliases such as "anywhere" are ignored. Panda revision 20 can recall compatible simulation revision 4 through 19 recipes within the same owner scope; all replayed actions are revalidated. Other arm packs remain isolated.

7.5 Joint motion followed by Cartesian work

All seven Panda articulations retain their native joint limits and can move the TCP beyond the smaller Cartesian task box. Before a later pick, place, cube turn, gesture, drawing, writing, home or Cartesian move, the simulator detects that condition. It preflights and executes a measured joint trajectory back to the task-ready posture, retaining any grasp, then runs the task. The record includes `cartesian_recovery` evidence. It does not reset the scene or teleport an object. A collision or invalid carried-object path still stops the transition. Fixed-TCP tool rotation does not silently reposition the TCP to recover.

8 Demo memory, lifting and quiet curses output

8.1 The recommended setup stays simple

Keep Celestia Auto-reply ON, Aster Auto-reply OFF, and Wake Studio OFF for both. Keep all five processes running. Privately address Celestia for one coordinator; use the illustrated Guided setup when a shared Conversation Studio room is needed. A request to choose a demonstration permits a bounded sequence for that request, not an endless exchange of new tasks.

8.2 Lift the cube and let it fall

The Panda simulation pack now advertises `lift`. It picks a cube if necessary and holds the bottom of the cube at a chosen height above the table. Opening the gripper then lets the simulator's gravity act. This is distinct from lowering an object onto a support with `place`.

```
Lift the red cube to 15 cm above the table then release it.
Get the blue cube then lift it then open the hand to let the cube fall.
```

Send either sentence privately to Celestia. Direct Aster/Studio task entry also accepts it. The default lift height is 0.15 m; the permitted range is 0.05–0.25 m, measured from the table to the cube's bottom. This is an absolute height, not an additional movement distance. The demo lift supports cubes, not the pen. If Celestia has asked for height, "you decide" keeps the same cube and release outcome and chooses the documented default.

The task record includes the measured lift height, retained grasp, release and landing positions, fall distance, final support and a stable settling window. The release does not teleport the cube or reset its velocity. A completed task requires measured success for every step. Workspace, collision, grasp and cancellation checks remain active. These new demonstrations are exposed only by the Panda simulation pack; no physical Panda driver is added.

8.3 Macarena-style movement and reusable names

```
Dance macarena.
Clear the drawing area then draw a circle then do your macarena.
Learn "my finale" by doing "draw a circle then dance macarena".
Clear the paper then perform "my finale".
List learned skills.
Forget learned skill "my finale".
```

The Macarena is an eight-beat *single-arm adaptation*: alternating reaches, wrist orientations, a low sweep and a return to rest. It is not a human full-body dance. A held pen is parked before the routine; a held cube is returned to its observed resting position when that origin is known. If it is not known, Aster requests a placement instead of guessing.

Successful measured execution makes the name "macarena" available in the current skill scope. Explicit `Learn "name" by doing "recipe"` runs the recipe once and commits the name only after complete verified success. A failed, cancelled or partially completed task is not a learned success. A named routine can be one step inside a longer task; the controller loads its recipe, resolves live objects and checks the current pack again. Recursive skills and plans exceeding 12 actions are rejected.

The arm's planner receives the saved skill names for this requester, while Celestia retains verified names returned for her requesting human. Aster keeps the full recipes in its local SQLite ledger.

Skills are scoped to robot, requester, arm pack and delegated conversation; another human does not inherit private recipes through Celestia. An explicit active-memory reset starts a separate delegated conversation scope. Use the same requester route when teaching and recalling a routine.

Preserve the runtime when upgrading. The default ledger is under `runtime/robotics/robot-arm/`. Use the included in-place updater to preserve runtime, personas and settings. If you extract the complete release into a new directory, stop all old processes and copy the old `runtime/` directory into the new project before restarting. A different runtime directory starts a different ledger. Back up before copying; never run two adapters against the same ledger.

Panda revision 12 can read compatible revision 4 through 11 recipes in the same owner scope and revalidate them. This cannot recover an earlier routine that was never saved as a verified recipe. The provided Macarena adaptation gives that name a concrete, repeatable implementation going forward.

8.4 Show or hide system messages in curses

Type one of these commands in Matteo's curses input, then press Enter:

```
:system_messages=off
:system_messages=on
:system_messages
```

Command	Effect
`:system_messages=off`	Hide new system notices, robot-camera receipt notices, send notices and Studio floor notices in this curses window.
`:system_messages=on`	Show those notices again. This is the startup default.
`:system_messages`	Display the current setting without changing it.

Human and AI chat and structured terminal task/call records remain visible. The command changes display only: cameras, capability discovery, guardrails, robot commands and logging continue. It does not erase existing scrollback and does not change the HTML5 interface. The setting lasts for this client session; use ON when diagnosing a connection problem.

8.5 If a send response is late

The client now allows up to 120 seconds without an HTTP response while the server completes Conversation Studio processing; human input still uses the asynchronous send queue. The timeout can be configured with `MMSW_SEND_READ_TIMEOUT_S` between 5 and 300 seconds. A read timeout explicitly reports *delivery unconfirmed* and its exception type instead of a blank "Could not reach" message. The client does not automatically resend. Check the task feed before repeating a movement, because the server may already have delivered the original command.

9 First verified cube rotation

9.1 Studio control

Select Panda and wait for fresh telemetry. Select `blue_cube`; inspect its received position and yaw. Under arm rotation, enter `-70` and press **Pick, turn & park cube**. The cube is grasped, lifted, rotated, lowered and released. Its top arrow indicates the measured object heading.

The scene yaw field remains useful for arranging a simulated test scene. **Set scene yaw** changes an object's simulated initial orientation while the arm is idle. It does not prove that the arm can grasp or rotate it. A successful arm rotation reports `arm_moved: true` in its rotation evidence.

9.2 Copy-and-send commands

These are direct Aster command forms, for an intentional direct command route or the Studio task input. In the recommended Celestia-led workflow, privately send the task wording to Celestia without the `#robot_arm` prefix, as in Section 6. Do not submit the same example through both routes.

```
#robot_arm Rotate the blue cube -70 degrees in place
#robot_arm Get the blue cube then rotate it a little then park it again
#robot_arm Get the blue cube and rotate the base just a little and park it again
#robot_arm Rotate the blue cube 30 degrees clockwise in place
```

"In place" means the resting location is preserved: the arm lifts the cube before turning it. It does not drag or twist the cube against the table. "A little" and "a bit" are documented as a relative positive 15-degree turn unless a direction is given. Positive world-Z rotation is counterclockwise viewed from above; clockwise is negative.

"Rotate the base" is accepted as a cube turn only when a cube is already held and the wording does not explicitly name the robot's base joint. Without a held cube, a base request selects joint 1. Explicit robot base or base joint always selects joint 1. Name the colour if several cubes are present. No arbitrary choice is made for "the cube" without a unique current reference.

9.3 Hold after rotation or put it elsewhere

Without "in place", a natural cube rotation retains the grasp. Follow it with a destination or a request to park:

```
#robot_arm Get the blue cube then rotate it -70 degrees then park it again
#robot_arm Rotate the blue cube 15 degrees then place it in the free safe zone
```

Parking a held cube uses its measured earlier resting position. Pen parking is a separate named fixture. If a held cube has no known resting origin, specify a validated destination instead of expecting the controller to invent one.

9.4 Teach "Cube rotation"

```
#robot_arm learn "Cube rotation" by doing "Rotate the blue cube -70 degrees in place"
#robot_arm Cube rotation
#robot_arm list learned skills
#robot_arm forget learned skill "Cube rotation"
```

Teaching runs the recipe once now. The name is stored only after measured completion and returned camera observations. The stored recipe is relative: each later invocation turns the then-current orientation by another -70 degrees. It is not an absolute heading command. Stop, connection settings and device permissions cannot be redefined as learned skills.

10 Tool orientation and coordinate conventions

Value	Convention
XYZ position	Metres in the advertised state frame. Panda simulation uses a right-handed world frame, Z up; X forward and Y left.
Joint angles	Radians; joint limits come from the selected model.
`angle_deg`	Signed relative degrees. Each rotation action permits magnitudes 0.1 through 180 degrees. Reach checks can reject a value within that range.
Quaternion	Unit `[x,y,z,w]`, accepted within 0.01 of unit norm and normalized. Zero, nonfinite and malformed values are rejected.
`world` axis	Fixed axes of the current world frame; the delta rotation premultiplies the current orientation.
`tool` axis	Local axes of the current TCP; the delta rotation postmultiplies the current orientation.

For orientation matrix R and incremental rotation R_Δ:

$$R_{\text{new}} = R_\Delta R \quad \text{(world axes)}, \qquad R_{\text{new}} = R R_\Delta \quad \text{(tool axes)}.$$

For a downward-pointing gripper, positive local Z and positive world Z can turn in opposite directions. Always state the frame when that distinction matters.

```
#robot_arm Rotate the gripper 15 degrees around world z
#robot_arm Rotate the gripper 20 degrees around tool x
#robot_arm Rotate the gripper -20 degrees around tool x
```

These actions keep the TCP position fixed, rather than keeping a held object's centre fixed. The cube-specific operation maintains the lifted cube's centre while rotating it. A general tool tilt may leave a cube unsuitable for a flat table placement; restore an upright grasp before placing it. A pen also needs its upright contact/parking configuration for writing.

10.1 No unlimited spindle rotation

The Panda has bounded joints. A full 360-degree sweep can require reconfiguration or regrasping, which this release does not automatically plan. Multiple bounded requests remain subject to live reach, joint and collision checks. Do not interpret the existence of three-axis control as guaranteed reach for all orientations at all positions.

11 All seven Panda articulations

Panda has seven revolute arm joints. The `libfranka::JointPositions` interface accepts seven desired joint angles in radians, ordered J1 through J7. These joints rotate over finite ranges; none is exposed here as an unlimited spinning axis. The gripper fingers still open and close linearly.[1, 7]

Joint	Operator label	Minimum (rad)	Maximum (rad)
J1	Base	-2.8973	2.8973
J2	Shoulder	-1.7628	1.7628
J3	Upper arm	-2.8973	2.8973
J4	Elbow	-3.0718	-0.0698
J5	Forearm	-2.8973	2.8973
J6	Wrist	-0.0175	3.7525
J7	Flange	-2.8973	2.8973

These are the published FER/Panda position limits, intersected with the loaded model limits. They are not FR3 limits. Joint targets reserve 0.001 rad inside each endpoint. Collision or carried-object workspace checks can reject a target inside these ranges. Positive angles follow the joint's own URDF axis, not an observer camera direction. J6 can legitimately exceed 180 degrees in an absolute pose.

11.1 Natural commands

```
#robot_arm Rotate joint 3 by 10 degrees
#robot_arm Rotate J7 -15 degrees
#robot_arm Rotate the robot base 15 degrees
#robot_arm Rotate the elbow -10 degrees
#robot_arm Get the blue cube then rotate joint 7 by 10 degrees then park it again
```

The robot base is J1; the elbow is J4. Specify a number for ambiguous shoulder or wrist articulations. A bare "rotate the wrist" retains the existing tool-orientation interpretation; say "joint 6" or "joint 7" for the mechanism. Explicit "base joint" always means J1, including while holding a cube. The earlier "rotate the base" shorthand still means cube yaw when a cube is already held.

11.2 Studio controls

The observer view labels all seven joints by number and name: Base, Shoulder, Upper arm, Elbow, Forearm, Wrist and Flange. Labels follow the rendered joint positions, with leader lines separating nearby joints. Check **Hide labels** to hide joint labels together with object and parking labels; uncheck it to restore them.

The **Panda articulations** card appears when the active simulation pack advertises the operation. Select a joint and enter a signed relative angle in degrees. The card displays the measured angle and its limits. Open **Set all seven joint angles**, use **Copy current measured angles**, edit the targets, then submit. The UI uses degrees and converts to radians for the absolute API. Other viewers receive the actual articulated pose through MMSW. A scene edit is not used to fake this movement.

11.3 Typed commands and verification

```
{"op":"rotate_joint","joint":3,"angle_deg":10}
{"op":"joints","positions":[0,-0.45,0,-2.25,0,3.2,0.7854]}
{"result":{"goals":[{"op":"rotate_joint","joint":4,"angle_deg":-10}]}}
```

The first command turns one joint relative to its measured starting angle. It accepts 0.1 to 180 degrees in either direction, subject to the remaining joint range. The second specifies all seven absolute angles in radians and excludes finger opening. It demonstrates that a legal J6 target can

exceed π. The third is the planning-provider schema. Learning a joint-turn phrase stores a bounded, verified recipe; it creates no driver or permission.

Before movement, a separate collision world samples the entire proposed joint path without changing the live arm. The real simulation tracks a quintic joint reference using motor dynamics. Reference limits are 0.55 rad/s, 1 rad/s^2 and 8 rad/s^3; live contacts and cancellation are checked during execution. A carried object's bounds and collisions are checked as well. Grasp constraints remain a simulation approximation.

A successful joint record contains the seven measured starting, target and final angles, canonical joint names, path-check status and held-object state. Every final joint must be within 0.006 rad of its requested target and the held state must agree. The controller rejects incomplete or inconsistent joint evidence. A collision, cancellation or limit error cannot count as a completed turn.

Joint motion can move the TCP and rotate a carried object about an arc. To keep the TCP position fixed, use `rotate_tool`; for cube yaw about its lifted centre followed by park-back, use `rotate`. No automatic obstacle detour, singularity-avoiding regrasp or unlimited 360-degree spin is added.

11.4 Physical Panda alignment

The command order, units, names and position limits match the Panda joint-position interface. This alignment does not commission physical motion. The future hardware driver must use measured robot state, the installed robot's limits, a real-time FCI loop and its own trajectory/contact handling. Do not replay the simulator reference as a hardware control loop. Physical Panda motion remains disabled in this distribution.

12 Executable action reference

Studio's structured-action control accepts JSON. All fields are validated before dispatch. A plan contains at most 12 bounded task actions. Individual motion operations execute sequentially; scene changes and connection management are separate controls.

12.1 Cube turn

```
{"op":"rotate","object":"blue_cube",
 "angle_deg":-70,"place_back":true}
```

`object` must be a known cube. `angle_deg` is relative world-Z yaw. `place_back` defaults to false for direct actions: the cube stays held. True returns it to its original resting centre and verifies an empty gripper and settling. A different held object blocks the request.

12.2 General tool turn

```
{"op":"rotate_tool","axis":"z",
 "angle_deg":15,"frame":"world"}
```

Axis and frame are mandatory in structured input. The tool position stays fixed within the motion tolerance. The command retains the current payload. It does not implicitly pick an object or release the fingers.

12.3 Full Cartesian pose

```
{"op":"move","position":[0.42,0.0,0.40],
 "orientation":[1.0,0.0,0.0,0.0]}
```

This example describes the downward tool orientation. Use the quaternion for the intended pose; do not put Euler angles into the quaternion field. When orientation is omitted, a held payload retains its carry orientation. An empty tool uses the existing downward default. This pose action is a direct manual control, not a free XYZ goal for the language planner. The Panda pack supports it; other packs reject the added orientation field rather than silently ignoring it.

12.4 Existing task operations

Operation	Input and behavior
`pick`	`object`; approach, close and lift the identified object.
`gripper`	`width_m:0..0.08`; independent total finger opening. Zero closes without selecting an object.
`grasp_together`	Exactly two cube IDs in `objects`; contact-qualified simultaneous seam grasp. See Section 17.
`grasp`	`object`; close at the current pose after proximity validation; it is not an automatic approach.
`place`	`object`, `position:[x,y,z]`; object-centre destination in metres. Includes picking if needed. Optional `support_object` requires actual stable support after release. Preserves a carried cube's orientation.
`release`	Open the held grasp at its current position; verify supported settling. Use placement for a controlled return.
`stack`	Cube `objects` in bottom-to-top order at the pack's stack site.
`arrange`	Cube `objects` in the requested left-to-right order at the pack's arrangement sites.
`group`	Sphere `objects`; put them together with verified separation from cubes.
`draw`	`shape` is `A`, `cube` or `circle`. Includes getting the pen.
`write`	Literal `text`, 1–512 characters; wraps and fits the native paper.
`trace`	Validated paths, style and image hash; used by image or composed-vector drawing.
`gesture`	`name`: `wave_hello`, `wave_goodbye`, `nod` or `bow`. Retains a held payload.
`home`	Return to the pack's rest pose; place/release any held object first.
`observe`	Acquire current state without a task motion.
`clear_paper`	Clear simulated strokes with the pen lifted.
`reset`	Reset the simulated scene and arm. This is a scene reset, not a physical recovery procedure.

12.5 Semantic goals and planners

The reasoning proxy uses symbolic objects and destinations. For rotation:

```
{"result":{"goals":[
  {"op":"rotate","object":"blue_cube",
   "angle_deg":-70,"place_back":true}
]}}
```

This is the provider's strict response wrapper, not the direct Studio action. The local resolver converts it into the same native operation. Other semantic destinations include a safe region, an unoccupied safe region, paper, another cube or a known fixture.

Planner	Behavior
Offline	Deterministic supported wording, current held state and scoped learned recipes. No provider call.
Auto	Uses local understanding first; unfamiliar wording may use the configured MMSW provider with schema validation and review.
AI	Uses provider planning with complete-request checks and independent review. A provider cannot enable an unavailable operation.

The default robotics model is `gpt-5.6-terra`; `--model`, `--reasoning-model`, `MMSW_ROBOTICS_MODEL` and `MMSW_ROBOTICS_REASONING_MODEL` select configured provider models. Availability depends on the user's provider account. Built-in rotation needs no paid model. Do not put credentials in a shared manual, task, screenshot or source archive.

13 Motion execution and verification

13.1 Panda simulation sequence

1. Validate the action schema and current pack. Reject a stale expected pack binding, unsupported operation, invalid angle or unavailable object.
2. Resolve object identity and the original resting position from current state. Reject a tilted cube or conflicting payload. For park-back turns, check the final cube footprint against the safe zone and other objects.
3. Use the normal Panda pick routine: align the parallel fingers to the cube yaw, approach, close, attach the simulation grasp constraint and lift.
4. Form the relative world-Z rotation. Check sampled orientation targets against Panda IK and joint limits, then track bounded servo waypoints. Stop on cancellation or a detected collision.
5. If requested, lower the cube at the saved origin, release, retract with the same tool orientation and let the body settle under gravity.
6. Measure the final object quaternion, centre, held state and support motion. Require the returned post-task camera observation before committing successful completion and learning.

The grasp is constraint-assisted after proximity and closure checks. It is not a force-closure or frictional grasp validation. Rotation never edits the cube pose directly. Scene editing remains a different, explicitly labelled operation.

13.2 Numerical acceptance

Check	Current simulator criterion
Orientation	Quaternion angular error at most 3 degrees. Quaternion signs q and $-q$ represent the same orientation.
Park-back position	Final object-centre displacement from the saved origin at most 18 mm.
Tool position after turn	At most 6 mm from the pre-turn TCP position.
Settling	Supported body with low linear/angular velocity for a continuous 0.2 s window.
Settling speed bounds	Linear speed below 0.003 m/s; angular speed below 0.03 rad/s.
Grasp/release	The requested cube remains held for a retained turn; park-back requires an empty gripper.
Evidence	Expected and measured quaternions, signed angle, axis/frame, error, positions and provenance. A generic success flag is insufficient.

These are application verification thresholds, not real-robot accuracy specifications. Measured errors in the supplied regression evidence are typically much smaller. The unit/regression and live-transport reports describe the scenarios actually tested.

13.3 Evidence versus camera observations

Simulator coordinates and quaternions are ground truth from PyBullet. The sensing images are separate observations that pass through MMSW. A camera can confirm visible anchors and provide visual context, but a single colour image does not measure physical depth, contact force or a cube's full calibrated pose. A visually symmetric cube also needs a heading mark or a calibrated pose target to disambiguate a physical rotation.

The observer camera, the robot's sensing camera and the pinned final-task image have different roles. Orbiting the observer view changes the local display. Simulation sensing-camera presets change shared robot observations. A real fixed camera cannot be moved by selecting a virtual view preset.

13.4 Records and retained state

The default adapter runtime is `runtime/robotics/robot-arm`. Its task ledger stores correlated records and verified recipes; observation files provide before/after evidence. Command IDs are idempotent: replaying the same ID does not issue a second movement. The same ID with conflicting contents is rejected.

A completed turn adds quaternion evidence and final object orientations to the task result. Celestia should report that measured outcome rather than infer success from a requested plan. Result-image sharing retains the .62 image-upload behavior: missing upload credentials do not manufacture a public image URL or change the motion result.

14 Stop, recovery and shared control

One active task owns motion. Other commands may wait in the bounded queue; passive viewers receive the same scene. Stop/cancel, stale required sensing or requester disconnection cancel work. Reconnection does not silently resume a previous movement.

```
#robot_arm Stop
#robot_arm Observe
```

A stopped arm attempts to hold its current configuration and retain an attached payload. Inspect the measured held state before issuing the next task. If a cube remains held after a cancelled turn, park it or choose a measured placement destination. Do not assume that cancellation restored its original angle.

A dropped body continues to move under gravity until it reaches support. Wait for settling before picking again. A pen that has fallen or tilted cannot be used as though it were upright; in simulation, restore it using the explicit scene/fixture controls. On hardware, use the commissioned recovery procedure.

15 Sharper task views and open-ended pen work

15.1 The recommended setup remains the same

Keep Celestia's Auto-reply ON, Aster's Auto-reply OFF and both Wake controls OFF. Keep all five processes running. Send a task privately to Celestia, or use the Guided shared-room setup in Appendix A. No new Agency or Expert-mode change is required. Aster advertises the updated SDK and Panda simulation revision 12 through the hidden capability exchange.

15.2 Watch the work without streaming high-resolution video

Open Robotics Studio and use the main **Observer view**. This view renders received joint/object poses on your computer. It follows measured feedback; interpolation smooths the display between packets and is not a new measurement.

1. Under **Observer motion**, choose **Angle slideshow** for changing angles, **Slow orbit** to circle the arm, or **Drawing sequence + final zoom** for pen work.
2. Set **View quality** to **Sharp – local rendering**. Use **Economy** if your computer's graphics become slow. Sharp allows up to twice the CSS pixel resolution where the display supports it; it adds local GPU work, not extra camera traffic.
3. Click **Expand view** for a larger workspace. Use **Drawing close-up** to look straight at the paper, or the **Overview**, **Top** and **Front** buttons.
4. Leave the small sensing-camera preview running. It remains independent of your observer angle and supplies the before/after observations used by the controller.

The curses client reports progress and returns an image link. It cannot show the interactive 3D observer in the terminal; open Robotics Studio alongside curses for live visual follow-up. Conversation Studio coordinates speech and does not provide this 3D view.

View or stream	What it contains	Load behavior
Sensing preview	Default 256 × 192 simulated camera; real-camera source remains separate	Default 1 frame/s; live robotics packet limit remains 96 KB.
Observer view	Local 3D render of received poses; selectable angles and paper close-up	Existing pose stream; Sharp changes only local rendering. Changed ink keyframes are capped at 2/s, with periodic recovery frames.
Final task sheet	Up to 1600 × 1200; overview, front, top and orthographic paper detail	One PNG in a successful simulation task result. Image cap 220 KB; terminal record cap 512 KB.

These are size and rate bounds, not a guarantee of zero load or latency. Each recipient still receives its own result; slow connections reduce live update rate instead of accumulating a video queue. Dense drawings may use a simplified live preview. The final sheet uses the recorded strokes and may reduce its resolution to fit its image budget.

15.3 Open and save the finished views

After a successful simulation task, Celestia/Aster's normal **Task result image** link points to the new sheet. In Robotics Studio, a result addressed to that browser enables **Latest task views**:

click **Open full-resolution sheet** or **Save PNG**. A task sent privately through Celestia keeps its result private to that route; another watching browser sees live motion but does not acquire that private task's record.

The sheet uses one pinned measured simulation scene. It does not move the arm, change the shared sensing-camera viewpoint, or render the proposed drawing as if it had been executed. The paper detail contains actual recorded pen paths. It intentionally omits arm/object occlusion and is labelled as an orthographic ink view. The other panels retain the simulator's wireframe appearance; this update adds resolution and viewpoints, not a photorealistic or physical camera.

The original after-task sensing observation remains in the task record. A task sheet is labelled `measured_simulation_multiview`; it is separate from `rendered_wireframe_camera` or `physical_camera`. The physical camera is not replaced with synthetic alternate views. If sheet creation fails, the sensing frame remains available; if external image upload fails, the measured task result still arrives and the captured image remains in task history.

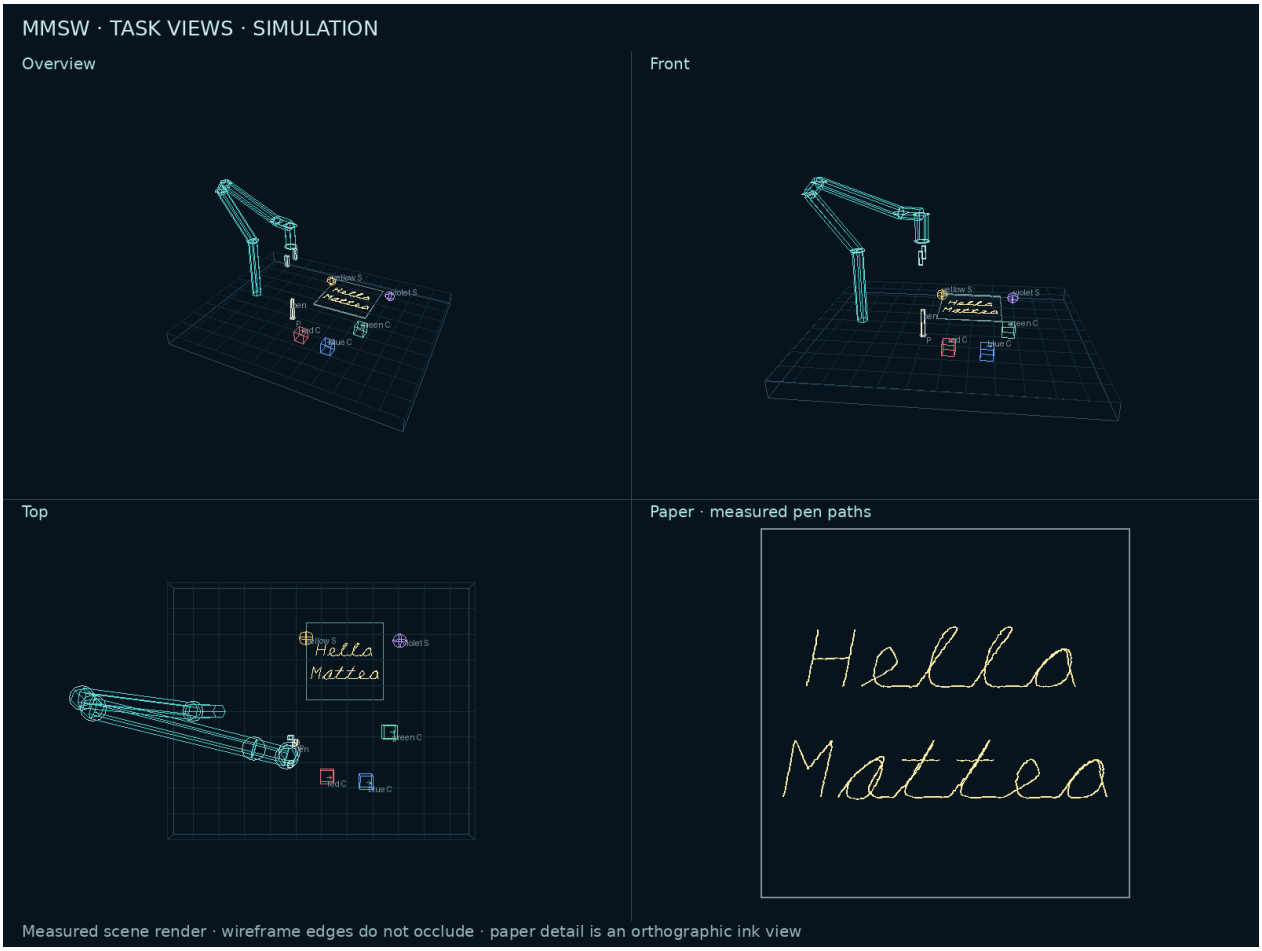


Figure 1: An actual simulated cursive writing result. The arm traced "Hello Matteo" and parked the pen. The sheet uses measured scene geometry and recorded ink. This example's PNG is 33,896 bytes; future tasks may differ.

15.4 Ask for a subject and style in ordinary language

Use **Auto – reasoning proxy** in Robotics Studio, or ask Celestia normally. The configured provider interprets the entire request and can select a general `artwork` goal: subject/composition, a free-form style description, and a detail level. A separate design call produces bounded vector

geometry, which is validated and independently reviewed before any motion. There is no list of permitted subjects or artist names in the drawing executor.

```
Clear the paper, then draw a detailed monkey at a typewriter
in a fine pen-and-ink style, then park the pen.

Draw a precise mechanical garden as a plotter would draw it.

Write "Hello Matteo" in flowing cursive inspired by
Renaissance manuscript lettering, then park the pen.

Write "MMSW DEMO" like a machine, with precise block strokes.
```

Send one example at a time. Style references guide appearance; they do not guarantee an exact artist replica or independently verified visual likeness. For an exact inscription, quote the words. A request such as "write in cursive" without any words may correctly require one clarification. Instructions such as "like a machine" must not become the literal inscription.

The planner can compose contours, ellipses, cubic Bezier curves and hatching. The native pen supports constant-width lines: tone and texture come from spacing and line placement. There is no pressure control, painted fill, variable ink colour or unlimited detail. Prefer visible features around 4 mm or larger on the 300 mm square Panda sheet. Fine resemblance depends on the generated design and the arm's measured accuracy.

15.5 Plotter and cursive writing without AI interpretation

In Studio's **Write on the paper** card, type the words, choose **Machine / plotter (default)** or **Cursive**, and click **Pick up the pen and write**. Plotter uses the existing original block centre-lines; Cursive uses an original connected lowercase alphabet, with separate dots/crossbars and slanted capitals. Both fit and wrap the text on the paper. Latin letters, common accents, digits and supported punctuation are supported. Build .70 raises the limit to 512 characters and uses a 6 mm minimum cap height; a layout that cannot fit is rejected rather than truncated.

The native structured form is:

```
{"op":"write","text":"Hello Matteo",
 "style":{"family":"cursive","slant_deg":12,
          "width_scale":1.0,"spacing":1.0}}
```

Omit `style` to use machine/plotter lettering. An unspecified drawing style likewise defaults to machine/plotter; a requested style is preserved in the design brief. Supported parameter bounds are family `plotter|cursive`, slant −25 to 25 degrees, width scale 0.65–1.35, and spacing 0.7–1.4. Custom letterforms can instead be composed as vector sketches. These parameters change actual pen paths, not just the reply text or final image.

15.6 Design limits, memory and evidence

A generated sketch has at most 120 primitives. Each primitive is a polyline, an ellipse or a cubic Bezier. Beziers are flattened locally; normalized points must remain inside the paper. Native traces allow at most 160 strokes and a complete task allows at most 4000 trace waypoints. A plan still has at most 12 actions and a vector trace has a 900-second execution budget. Longer writing has a length-dependent budget capped at 3600 seconds. Rejecting malformed or excessive geometry does not disable reasoning: the provider can revise one rejected proposal. More detail may require separate tasks/sheets.

A verified drawing recipe stores the actual geometry. Repeating a learned name therefore reuses those paths without redesigning the subject. For example:

```
Learn "my garden" by doing "draw a mechanical garden".
Clear the paper then do "my garden".
```

Preserve the runtime directory and the same requester/conversation route when upgrading. Compatible Panda revision 4 through 11 recipes are available under revision 12 and are revalidated. The update cannot recover a recipe that was never saved.

Execution checks the measured pen contact, stroke count, path error, paper bounds and final pen-up state. The design metadata identifies the requested subject/style and hashes the geometry. It does not prove that a viewer recognises the subject or that the result matches a named artist. Live paid-provider artistic quality and physical Panda pen skills are not validated by the local regression suite.

16 Contact control, fitted writing and saved drawings

16.1 What changes in build .70

Panda simulation revision 12 retains two physical interaction primitives: a closed-fingertip `push` and a short `throw`. Celestia learns their current fields through Aster's hidden capability exchange. The general planner can select them from the whole request and the measured scene. It must preserve whether you asked for contact, placement, release or a moving release; a pick-and-drop is not verification of a push.

Keep the simple setup from Appendix A: Celestia Auto-reply ON, Aster Auto-reply OFF, both Wake controls OFF. Conversation Studio remains optional. In a shared room, use Guided mode, Celestia as the fixed primary host, and Aster's Session role **Excluded**. This excludes conversational turns, not delegated robot execution. No Expert setting is needed for these new skills.

After upgrading, restart the server, Robotics Studio, Aster, Celestia and Matteo from the same build and refresh the Studio page. The runtime identity should show v59.20260919.91 and the Panda pack revision should show 20. Preserve the runtime directory to retain learned routines and saved sheets.

16.2 Push a cube using contact

Send the following tasks one at a time through Celestia. Wait for the measured terminal result before issuing the next task. These natural examples require the configured reasoning provider; they are not new literal phrase-matching rules.

```
Celestia, ask Aster to stack the red, blue and green cubes.

Push the top cube off its support with the closed gripper
fingertips, and let it fall onto a clear part of the table.

Push the red cube 6 centimetres along a clear direction.
```

Aster observes the cube's actual supporting body. It approaches an available face with an empty closed gripper, drives a bounded contact path, withdraws, and measures the final settled object. If a requested lane is obstructed or unreachable, it reports that limitation; it does not silently replace a push with a grasp. The slower edge mode allows the object to settle between small waypoints.

Push mode	Intended result	Verification
`slide`	Move a cube by a requested planar distance, normally on the table	Fingertip contact and measured displacement, followed by settling.
`edge`	Move the cube towards the edge of its measured supporting object while keeping support	Contact, bounded travel and the same supporting body after withdrawal.
`off_support`	Dislodge the cube from a supporting object into a clear tabletop landing lane	Contact, loss of original support, measurable fall and settling.

The support-edge modes are for a cube on another object; they do not push a cube off the outside of the table. A cube supporting another object must be cleared from the top first. The supported push target is a cube, not a sphere or the pen. Final displacement is measured and can differ from requested travel because of friction, tipping and settling.

16.3 Short throws and their measured limits

A `throw` is a short motor-driven tabletop toss in this simulator. The arm picks up a cube or sphere, finds an available launch lane, moves the held object, and releases its grasp while the measured body velocity is nonzero. Gravity and contact determine the flight and landing. It does not assign a new object position, inject an impulse, or reset an object's velocity to manufacture a result.

```
Celestia, give the red cube a short toss into a clear part
of the table, then tell me its measured landing position.
```

The current contract requests a servo speed from 0.15 to 0.45 m/s (default 0.30 m/s), not a guaranteed release speed or range. The controller returns the actual release velocity, flight time, displacement and settled support. There is no exact-target ballistic solver, long-range throwing, catching, pen throwing or physical Panda implementation in this release. A resting cube can be picked again after landing on any of its six faces. These simulation results do not establish hardware readiness.

16.4 Structured reference and general reasoning

The executor accepts a world-XY direction vector or `"auto"`. Auto selects among measured clear candidate lanes. A slide requests 0.02–0.16 m of travel. Edge/off-support travel is derived from the actual supporting geometry; the distance field does not override that geometry. The following structured actions are examples for integrators, not additional Guided UI settings:

```
{"op":"push","object":"green_cube","direction":"auto",
 "distance_m":0.06,"mode":"off_support"}

{"op":"push","object":"red_cube","direction":[1,0],
 "distance_m":0.06,"mode":"slide"}

{"op":"throw","object":"red_cube","direction":"auto",
 "speed_m_s":0.30}
```

The provider receives operation schemas, geometry and limits. It composes supported operations; there is no list of special synonyms for the reported conversation or a canned answer for each scenario. Relative references use the measured scene and conversation context. A later task that depends on an object's landing should observe its actual result rather than assume an exact landing coordinate. Cancellation remains effective during native motor steps; it does not reverse completed motion or suspend a released object in mid-air.

16.5 Machine is the default; longer text fits the sheet

If no style is specified, writing uses the machine/plotter alphabet and artwork receives a machine/-plotter design brief. Explicit cursive or other style requests still take precedence. Quote the intended inscription so the planner can keep the words separate from a style instruction.

In Robotics Studio, open **Write on the paper**, enter the complete text in **Text to fit on the paper**, leave **Machine / plotter (default)** selected, and click **Pick up the pen and write**. The layout engine wraps words and reduces letter size to fit the native paper margins. Existing explicit line breaks are preserved. It does not summarise, silently crop or discard the end of an inscription.

```
Clear the paper and write "Matteo and Celestia work together.
Aster writes this longer message on one sheet, reducing the
letter size and wrapping every word to fit." Then park the pen.
```

The Panda text budget is 512 characters, at most 32 explicit lines and a minimum cap height of 6 mm. The normal sheet is 300 mm square with inset margins. Geometry is bounded to 1600 lettering strokes and 40,000 lettering points; already accumulated ink also has a paper budget. Unusual line breaks or style parameters may hit those bounds before the character limit. Unsupported characters or an unreadable fit produce an actionable error. Clear the current paper before a separate composition; clearing does not delete the saved gallery.

The native sample below contains 162 measured strokes. Its maximum recorded path error is approximately 3.0 mm. Longer text is smaller, so servo accuracy limits fine detail even when the geometry fits. Native execution of this sample and layout checks up to 512 characters were tested; this does not claim that every possible 512-character inscription or artistic design has been physically simulated.

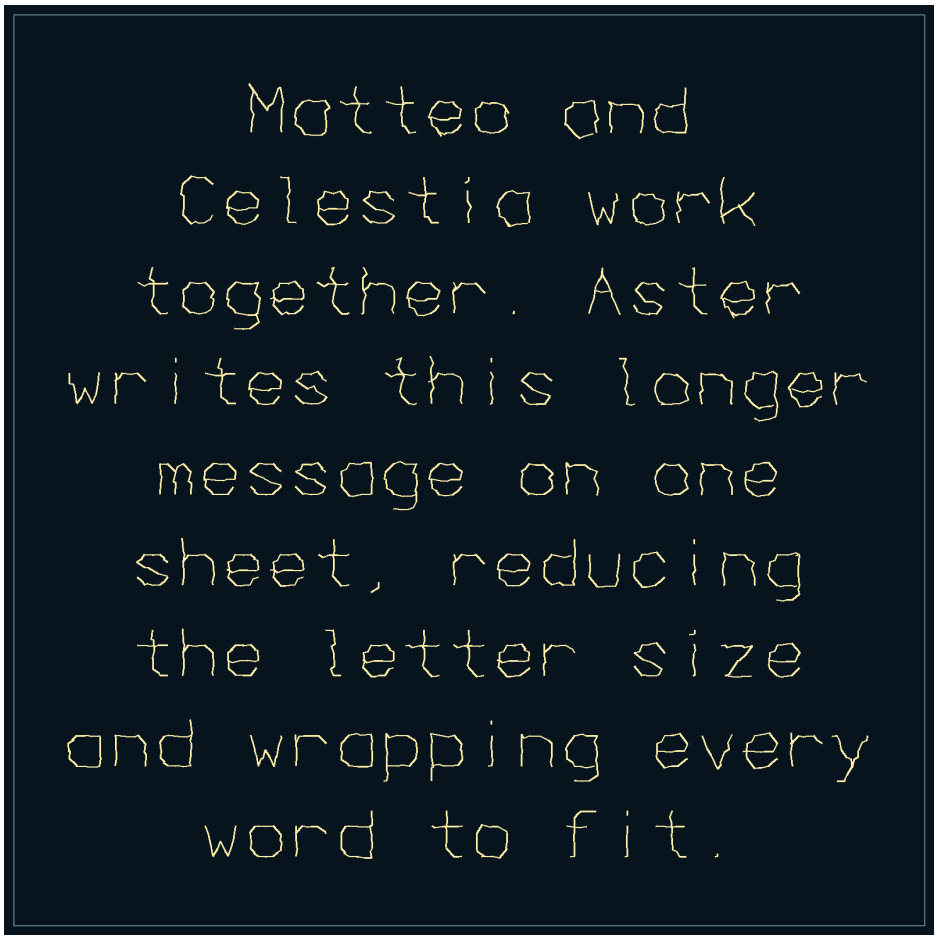


Figure 2: Actual recorded Panda pen paths, automatically fitted and wrapped. The 1600 × 1600 archived PNG is rendered on the same dark background as the task views. It is not a preview of unexecuted letter geometry.

16.6 Find and save every finished drawing

In Robotics Studio, expand **Saved drawings**. The count in the heading reflects the archive for the current robot runtime. The gallery appears in the same Studio page; Conversation Studio and Expert mode are not involved.

1. Finish a `draw`, `write` or `trace` step and wait for verification. The controller automatically saves the measured sheet as a 1600 × 1600 PNG.
2. Expand **Saved drawings**, or click **Refresh** if it is already open. Use **Newer** and **Older** to browse pages of up to 12 thumbnails.
3. Click a thumbnail to open its full saved sheet. Live camera updates do not replace this pinned image.
4. Click that card's **Save PNG** to download it. The stored title and timestamp identify the sheet.

Each verified drawing step saves the current measured paper. If a single task draws, clears, draws again and clears again, both completed sheets remain in the archive. A task that adds to existing ink saves the cumulative sheet. A later failure in the same task does not remove earlier verified sheets. Duplicate delivery of a command does not create duplicate archive entries.

The archive is in `runtime/drawings/`, relative to the configured robotics runtime, with PNGs, small thumbnails, metadata and an index. Scene reset, paper clear and process restart do not erase it. Back up and retain this runtime directory when extracting a new release. There is no automatic deletion of old sheets; disk use grows with the archive. The gallery starts saving with this release and cannot reconstruct paper that was cleared before saving was implemented.

Shared paper, private conversation. The gallery stores the robot workspace's visible measured drawings. Connected Studio operators can browse those saved sheets. It does not include requester identities, private chat transcripts or learned recipes. A private task's completion record still follows its original reply route.

16.7 Dark background and bounded MMSW traffic

The orthographic paper panel in a final multi-angle task sheet now uses the same dark background as the surrounding overview/front/top views (`#07141d`). Pale-gold measured ink remains visible against it. Archived drawing PNGs use the same colours; the old white/cream panel is no longer generated for new results. Earlier image links are historical files and are not recoloured retroactively.

While the gallery is closed, no thumbnails or full PNGs are requested. An open gallery refreshes when its small archive counter changes. The local Studio reads saved images locally. A remote Studio uses the existing MMSW operator command/result route to request a page or a single image; it does not contact an unexposed robot IP or add a second live image stream. Gallery reads do not execute motor actions.

The live pose and low-resolution sensing streams retain their existing budgets. A full archived PNG is fetched only when a user opens or downloads it. Remote full-image responses retain a 220 KB image cap; an exceptionally dense larger PNG remains available from the local Studio. This keeps the broker's terminal-message budget unchanged. The separate final task sheet remains at most 1600 × 1200 and may downscale to fit its own cap. The gallery provides the dedicated 1600 × 1600 measured paper view.

16.8 Quick checks after upgrading

Check	Expected observation
No style requested	Machine/plotter letter paths; an explicit style remains honoured.
Long quoted inscription	More lines and smaller letters, with the exact text retained.
Completed pen work	A new card in Saved drawings; saved PNG remains after Clear paper and restart.
Final task views	Dark paper detail matching the other panels, with visible pale-gold ink.
Push or toss	Native contact/flight evidence, not a narrated claim or substituted pick-and-place.
Aster Auto-reply OFF	Delegated commands still execute and return a measured result to Celestia.

If a drawing executes but storage fails, the task record carries an archive warning. Correct the runtime storage problem; do not assume a missing card means the arm failed to draw. If a newly introduced operation is rejected as unsupported, check that all running processes use v59.20260919.91and that the hidden capability manifest shows Panda revision 20.

17 Independent fingers, side grasps and balance attempts

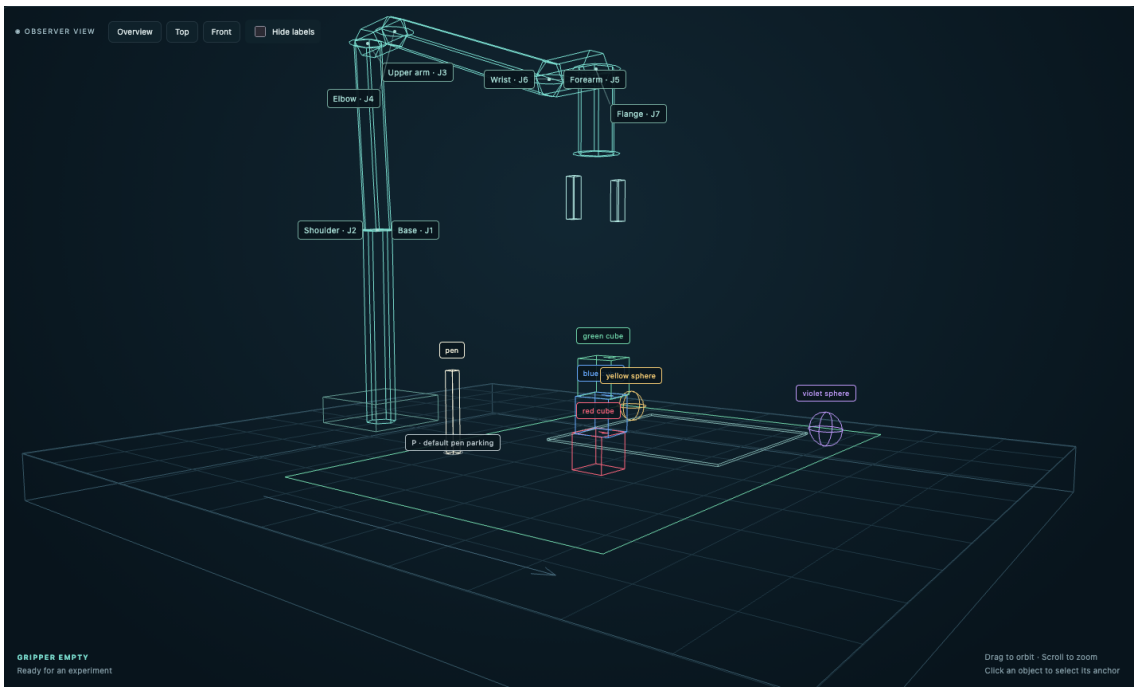
Build v59.20260919.91, Panda simulation pack revision 20, documentation revision 30. This section extends the current arm controls. The illustrated Guided setup in Appendix A still applies: Celestia Auto-reply ON, Aster Auto-reply OFF, Wake OFF. Aster and Robotics Studio must remain running.

17.1 What changed and why

An instruction such as “close the gripper and push the top cube off” contains two different physical operations. The previous pack exposed pushing but had no independent finger-opening action, so the complete request could be rejected before the push. It also tracked one grasped object, rejected a pickup whenever the target supported another object, and restricted named object-on-object placement to cubes.

The updated contract separates finger motion, grasping, contact pushing, carrying and release. Celestia receives the versioned SDK through Aster’s hidden capability exchange. She can preserve the requested approach and complete sequence instead of inventing a missing limitation or substituting a different physical method. The general provider handles wording and references; geometric checks and measured feedback decide what actually works.

The source findings explain real missing capabilities. They do not identify the build that was running in an unversioned conversation screenshot. Restart all five processes from this release, then confirm v59.20260919.91 and `panda.simulation`, revision 20, in the current arm status. Reconnect Celestia if needed and let the hidden manifest refresh. You do not need to paste this command list into her prompt.



User-provided observation from the reported failure: three stacked cubes below an open gripper. This original image documents the starting scenario; it is not evidence that the new operations ran.

17.2 Close or open the fingers without selecting an object

Say to Celestia:

> Just close the gripper.
> Open the gripper.
> Set the gripper opening to 40 millimetres.

In Robotics Studio, use the **Panda fingers** card, immediately before **Panda articulations**. Click **Close fingers** or **Open / release payload**. For an intermediate opening, enter **Total opening – millimetres** and click **Set opening**. The readout reports measured opening, not merely the target. No Conversation Studio Expert setting is needed.

```
{"op":"gripper","width_m":0}
{"op":"gripper","width_m":0.04}
{"op":"gripper","width_m":0.08}
```

The opening is the sum of both finger-joint displacements, from 0 to 80 mm. Contact or the retained payload can prevent the requested closure. Closing empty fingers never attaches a nearby object. Widening an existing grasp releases its complete payload and measures where it comes to rest. `grasp` remains a separate operation for a selected object already aligned inside the gripper.

For the original push request, a complete plan can close the fingers and then use `push` with `mode: off_support`. The push primitive also prepares its own closed fingertips. It measures contact, displacement and the final support; grasping and dropping are not substitutes for pushing.

17.3 Choose a side pickup, including a middle cube

With the cubes stacked red, blue, green from bottom to top, say:

> Get the blue cube from the side and keep holding it.
> Get the cube in the middle from the side.

Celestia resolves “middle” from the measured scene. Naming the colour also makes the intended selection explicit. The native pickup accepts `approach:auto`, `top` or `side`. In .79 the default first parks upper cubes, top-first, before extracting the selected cube. An explicit side approach remains a side approach after this preparation. When the requester explicitly accepts collapse and skips clearing, auto selects side extraction for the still-loaded cube. An explicitly blocked top approach is reported as blocked; it is not silently relabelled a top pickup after using a side approach. See Section 24 for the three clearing policies.

```
{"op":"pick","object":"blue_cube","approach":"side"}
```

For direct control, expand **Manual arm controls**, choose the **Selected anchor**, set **Pickup approach** to **From the side – cube**, and click **Pick object**. The arm selects a reachable side corridor, checks opposing finger contact, withdraws horizontally by about 12 cm, then lifts. Side pickup currently supports cubes; it is not a generic tool for every shape or an arbitrarily packed pile.

Only the selected cube is attached. With explicit permission to skip clearing and accept collapse, an upper cube can slide, fall, or ride on it through ordinary contact. Its final position and support remain observable, but it is not described as a second grasped object. Such an extraction does not guarantee that the rest of the stack stays unchanged; default clearing first removes that upper load.

17.4 One grasp of two touching cubes

For the upper two cubes in that same stack, say:

> Grasp the blue and green cubes together at their touching edge, in one grasp.

```
{"op":"grasp_together",
 "objects":["blue_cube","green_cube"]}
```

The operation requires two aligned, vertically touching cubes with a reachable shared edge. It checks that both finger pads contact *each* cube before attaching either. It then lifts both by about 7 cm and verifies their relative positions. Separate cubes, a blocked approach, missing contact, or an additional object resting above the requested pair can prevent completion. It never converts the request into two sequential pickups.

The simulator uses its existing **constraint-assisted grasp approximation** after contact qualification. This tests approach geometry, contact, movement and retention; it does not prove friction-only force closure or that a real Panda would securely hold two objects with those surfaces and loads. The measured finger opening is retained during transport.

`held_objects` is the complete list of attached objects. The legacy `held` field remains the first requested member for compatibility. Studio and Celestia use the complete list when reporting the payload. Opening or releasing detaches both objects and checks each object's support independently.

17.5 Keep holding while rotating

Say:

> Keep holding it and rotate the gripper 15 degrees around world Z.
> Keep both cubes held and rotate joint 1 by 5 degrees.

```
{"op":"rotate_tool","axis":"z",
 "angle_deg":15,"frame":"world"}
{"op":"rotate_joint","joint":1,"angle_deg":5}
```

Tool rotation supports world or local tool X/Y/Z axes. Joint rotation addresses J1–J7 and moves the tool along that articulation's trajectory. Both retain the current grasp, including a qualified pair, and check every attached object's collisions and relative pose. Neither implies a release or regrasp. An explicit Cartesian pose can also retain the payload.

The object-specific `rotate` operation remains a single-cube world-Z yaw operation. For a pair, use tool or joint rotation. Single-object placement cannot silently detach one member of a combined grasp: release the pair before asking to manipulate just one member.

Not every angle is reachable from every pose. Joint limits, workspace boundaries, surrounding objects and the size of the payload still apply. A rejected or interrupted turn retains the grasp where possible and reports the actual state. Success requires measured tool/joint motion and payload retention, not only an accepted request.

17.6 Try placing a sphere on a cube or the pen

Say:

> Try to place the yellow sphere on top of the red cube.
> Try to balance the yellow sphere on top of the pen.

The meaning-level goal preserves the named support:

```
{"goals":[{"op":"place","object":"yellow_sphere",
 "destination":{"on":"red_cube"}}]}
{"goals":[{"op":"place","object":"yellow_sphere",
 "destination":{"on":"pen"}}]}
```

These are planner goal objects, not standalone motor commands. The local resolver computes the object-centre target from measured dimensions and orientation. The executable `place` action includes `support_object`, which binds the required final support. When using Studio's Structured command box directly, submit a supported executable action or send the natural-language task in the normal task field.

After releasing, the sphere must remain supported by the requested object with low linear and angular velocity for a continuous **one-second observation window**. Position checks still apply. If it rolls onto the table, the requested balance failed, even if the sphere eventually comes to rest. The controller does not glue the sphere to the support or add a balancing fixture to make the test pass.

The pen starts in an explicit upright fixed-fixture approximation until first picked. Balancing on that initial pen therefore tests a fixed support, not a freely standing pen. After the pen has been picked it is dynamic; the same request can have a different outcome. Curved or narrow supports, friction, contact tolerances and small release errors affect the result. A one-second stable window is finite evidence, not a promise of indefinite equilibrium or hardware feasibility.

17.7 How to read a failed attempt

Reported condition	Meaning and next action
Missing capability / stale pack	Check the live build and revision. The hidden contract must advertise the operation and its fields. Reconnect after updating.
Clarification before execution	A reference or required constraint is missing. Answer the question; the complete pending task is retained.
Motion attempted; not verified	The executor was invoked. Some motion may have occurred. Inspect the current pose, held objects and final support before another task.
Contact or reach failure	The requested geometry did not satisfy the checked approach or motion. The task is not recorded as a learned success.
Balance failed	Inspect the sphere's actual support. Landing on the table does not satisfy placement on the pen or cube.

The result evidence distinguishes an attempted motion from a pre-execution clarification. An empty list of completed steps alone is not proof that nothing moved. Saying "you decide" preserves the physical method in the pending task; it must not change a push or throw into a lift-and-drop.

18 Spatial drawing and task reliability in .72

Required running build: v59.20260919.91, `panda.simulation` revision 20. A log showing .70 / revision 7 is still using the earlier controller. Restart the server, Robotics Studio, Aster and Celestia from the same updated installation. The invisible capability exchange then advertises the new contract automatically.

18.1 A simple workflow through Celestia

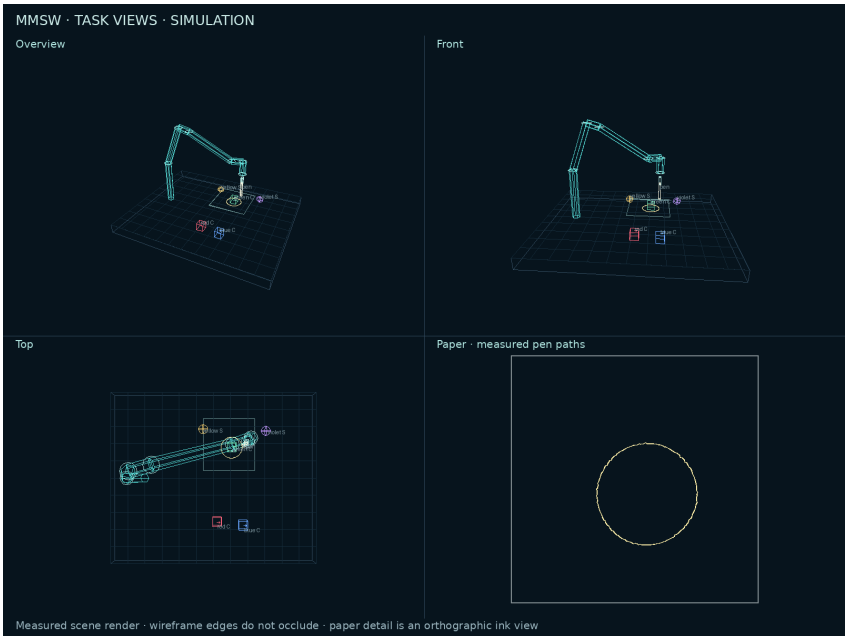
Keep the existing Celestia-led setup: Celestia Auto-reply ON, Aster conversational Auto-reply OFF, Wake Studio OFF. Keep Aster and Robotics Studio running, with Matteo–Celestia and Celestia–Aster connected in both directions. Conversation Studio remains optional; its illustrated Guided setup is unchanged.

Send one complete request, for example:

> Place the green cube slightly off centre on the drawing area, then draw a circle around it.
>
> Write “Hello Matteo” in the space that is still clear.
>
> Draw a small botanical study in the remaining blank area, using machine strokes.

Celestia delegates the whole intent. Aster places the object, measures its actual position, gets the pen and generates the drawing geometry. You do not need to supply pen-pickup instructions or a manually defined circular path. Other objects on the paper remain obstacles, even when the request names only one of them.

An unspecified style remains machine/plotter. Writing wraps and scales into the available area while retaining the inscription and its minimum supported letter size. Ordinary vector art keeps its proportions when translated and scaled. A full surrounding circle keeps its measured centre and required clearance; it is never silently shifted or shrunk across the object.



Native .72 simulation: a cube placed away from paper centre, followed by an object-centred measured circular pen path. This is an executed test scene, not a proposed illustration.

18.2 What “available space” means

Input or condition	Resulting behavior
An object on the sheet	Live world-space bounds reserve its footprint. Low objects also reserve pen-shaft clearance; taller objects reserve a conservative hand clearance.
Existing ink	Ordinary new drawings and text choose a blank rectangle that avoids the bounds of existing strokes. This is conservative: not every irregular gap is used.
Several objects	All object exclusion areas are considered. Only drawing inside the selected clear region is permitted.
Circle or rectangle around an object	The outline is computed from that object’s current measured footprint. Every complete segment must fit the sheet and clear all obstacles.
Travel between strokes	The pen rises vertically before crossing the scene and travels above measured object heights. Native arm collision checks remain active.
No usable region	The action reports the actual geometric limitation. It does not move obstacles, erase ink or replace the requested outline without a new instruction.
Object near the paper edge	Placing the object may fit even when a full circle around it cannot. Move it inward or ask for a smaller feasible composition.

The sheet is 300 by 300 mm. A placement destination may specify normalized paper coordinates `u` and `v`: zero is the lower world X/Y edge, one the upper edge. Coordinates describe the object’s centre; its entire oriented footprint must remain within the sheet. An unspecified paper placement starts at the centre and chooses another free position if occupied. Objects can occupy any feasible measured position; neither placement nor outline is restricted to a few named drawing zones.

The native SDK includes these examples for integrations:

```
{"op":"place","object":"green_cube",
 "destination":{"paper":{"u":0.55,"v":0.44}}}
{"op":"outline","object":"green_cube",
 "shape":"circle","gap_m":0.02}
```

These are semantic goals. The controller resolves placement coordinates locally. The `outline` native operation retains the object ID so it can measure the reference again immediately before execution. Circle and rectangle outlines currently support cubes and spheres, with requested gap 5–80 mm. Clearance may be larger to accommodate the pen and hand. Unsupported geometry still fails rather than inventing success.

18.3 Measured drawing and saved output

Completed drawings retain the dark background used elsewhere in Robotics Studio. Each verified `outline`, `draw`, `write` or `trace` step saves a 1600-by-1600 PNG in **Saved drawings**; clearing the scene does not delete that archive. The task also provides the high-resolution overview/front/top/-paper sheet. The ink-only PNG and the multiview task image serve different purposes.

Spatial evidence includes the selected free rectangle or reference footprint, measured pen bounds, obstacle displacement and a clearance-verification flag. Circle evidence includes centre, radius, radial error and closure error. This verifies geometry and measured pen motion, not artistic likeness or a particular artist’s style.

18.4 Rotation while retaining contact

“Rotate in place” describes the final location and may use the retained lift-and-return routine. An explicit “do not lift” instruction selects the separate `surface` method:

> Rotate the blue cube a little without lifting it from the table.
>
> Rotate the blue cube 15 degrees while it stays on the red cube; do not lift.

The rotating object remains blue in the second example; red is its support. The dispatcher and reviewer preserve those roles through follow-up requests instead of retargeting the support. When the user delegates a small unspecified angle, the default is 15 degrees.

```
{"op":"rotate","object":"blue_cube","angle_deg":15,
 "place_back":true,"method":"surface",
 "support_object":"red_cube"}
```

The simulator approaches the resting cube, grasps it without raising it, turns through joint-driven orientation waypoints, opens the fingers and retreats. Its existing constraint-assisted grasp approximation remains explicit. Every physics step during approach/grasp/turn/release records support contact. Success requires contact on every sample, at most 2 mm upward centre motion, at most 6 mm lateral centre drift, the correct final support and the normal orientation check. The release ends with an empty gripper.

A top-down surface turn currently requires a clear top approach and a measured table or cube support. An object above the selected cube can block that approach. The controller reports that limit without silently lifting, clearing the stack, switching targets or changing the requested method. This is bounded simulation evidence, not hardware force-control certification.

18.5 Fingertip push on a stack

An unspecified “push the green cube with the fingertip” uses `mode:auto`. On a supporting cube it chooses a small motion toward a stable support edge; on the table it chooses a 3 cm slide. Direction is selected from clear reachable corridors. The chosen travel and actual displacement are reported separately.

If you want the upper cube to fall, say so: “Push the green cube off its support.” That selects the retained `off_support` outcome. A specified 6 cm slide on a narrow support is not silently converted to a fall. Contact, displacement and settled support still have to verify. Failed verification can follow physical movement; it does not prove that nothing moved.

18.6 Curve tracking and clear failure messages

Vector drawing keeps the 4 mm path-error limit. Curve endpoints use matched IK/servo tolerances to avoid long convergence waits, and verification measures distances to continuous line segments instead of a sparse grid of comparison points. The old grid could add artificial error on long segments. A remaining failure states both the measured error and limit in millimetres instead of rounding both to an ambiguous “0.004 m.”

A failed attempt remains failed. Its record includes measured final object displacement and a current multiview scene when available. Partial ink stays visible and is not archived as a completed drawing. A retry is a new task; clear the paper first if you want a fresh sheet.

18.7 Why some tasks take longer to begin

Request path	Work before motion
Complete command recognized locally	Celestia can skip its dispatch-provider call when there is one authorized connected arm, a current capability manifest and a complete reference-free command. The normal graph and native motion checks still run.
Contextual or unfamiliar request	The configured provider interprets the whole request. Aster resolves semantic goals, validates the native plan and reviews coverage. A clarification is possible when a reference truly remains ambiguous.
Open-ended detailed artwork	A design pass generates bounded vector geometry before motion. Complexity and provider response time still affect startup; there is no fixed latency promise.
Verified remembered routine	Its stored recipe is expanded and revalidated against the current scene and pack. Learning does not bypass measured motion checks.

Local command dispatch reuses the complete existing parser; it is not triggered by finding an action word inside a longer sentence. Mixed requests and unresolved references continue through interpretation. Task records now expose observation/planning and first-action timing so delays can be located without weakening checks. Rendering the observer remains local; high-resolution task sheets and archive PNGs are produced per task rather than streamed continuously.

18.8 Reading results correctly

Reported concern	.72 correction
Throw succeeded but landing called unknown	Reply evidence explicitly retains measured settling and support before trimming long trajectories. A confirmed stable table landing can be described as such.
Reset succeeded but pen parking called unknown	The explicit current pen-parking status is retained, including after scene reset.
Push failed, therefore “nothing moved”	Outcome verification and observed displacement are distinct. A failed attempted motion must not be described as an unchanged scene without measurements.
Completed stack described as not learned	Replies avoid volunteering a missing saved routine when learning was not requested. Verified learning is claimed only when the ledger reports it.
New capability missing from Celestia	Check the running adapter build and pack revision, then reconnect/refresh. Hidden capability exchange replaces the old contract; no wake rule or pasted vocabulary is needed.

18.9 Validation limits

The release includes native PyBullet checks for an off-centre object outline, free-space lettering and vector drawing, paper-edge rejection, continuous-contact table/cube turns, support-aware pushes and the full place-then-outline controller path. Provider tests use controlled responses: they exercise schemas, routing and whole-task handling but do not establish the quality or latency of every live model response. Physical Panda commissioning remains outside this release.

19 Editing artwork already on the sheet (.73)

Panda simulation pack revision 12. This update addresses the reported follow-up in which a balloon request produced another cat, even though the requested cat was already on the paper. Aster now maintains a map of individual drawing operations, with a stable identity, creation order, measured position and ink bounds for each. Celestia preserves references such as "the first drawing" when delegating; Aster resolves them against the current sheet.

19.1 The simple workflow: keep the existing drawing

Use the same Guided setup described in Appendix A. Celestia's Auto-reply stays ON; Aster's conversational Auto-reply stays OFF. Wake Studio remains OFF. No Expert-mode setting is needed for paper awareness. Restart all five processes from v59.20260919.91 so Aster advertises revision 20 automatically through the hidden capability exchange.

Give the task once to Celestia. For the reported example, an unambiguous instruction is:

> Add a speech balloon around the first cat already on the paper, with its tail pointing toward the head of the other existing cat. Keep both drawings exactly where they are. Add only the balloon and tail.

The planner should bind the *contents* of the balloon to the first existing drawing and the *speaker* to the other drawing. These are two separate references. It should not ask the design provider to draw a new cat inside a new balloon. The corrected operation computes the outline from the first drawing's measured ink and adds one closed stroke, including the pointer when requested.

Request	How the current sheet is used
"Put a border around the first drawing."	Enclose the measured bounds of drawing 1 with a new outline; retain all original strokes.
"Use the first cat inside the balloon; point to the other cat."	Bind the enclosure and pointer to distinct existing drawing IDs. Aster uses the paper view and measured bounds to ground the reference.
"Add rays above that illustration."	Generate only the extra rays in fixed paper coordinates, with the existing drawing as an anchor.
"Draw another cat."	Create a new drawing and fit it into unused paper, because another subject was explicitly requested.
"Clear the paper."	Remove current simulated ink and create a new sheet identity. Previously saved PNGs remain in the gallery.

If it does not fit. Aster must report the blocked margin or ink crossing. It must not shrink, move, erase or redraw the original to make the request appear successful. On a tightly filled sheet, choose a smaller feasible addition or explicitly start a fresh sheet. A physical pen cannot relocate old ink.

19.2 What Aster remembers about the paper

The live `paper_content` map records each operation's identity, creation order, title, status, stroke range and measured bounds. A generated illustration and a later uploaded-picture trace remain separate entries even if both depict cats. Each addition references its original entries.

Titles come from the requested design or operation; they are not proof of visual recognition. An uploaded picture is labelled as an uploaded picture unless its content can be grounded from the actual paper view and conversation. If two possible references remain, Celestia or Aster should ask which existing drawing is intended. "First" means creation order; "left" and "above" refer to measured paper coordinates.

The planner and design step can inspect a 512-pixel paper-only raster of *measured pen paths*, together with drawing bounds and a bounded sample of the paths. This is generated locally when reasoning needs the paper. It does not increase the live camera stream rate or stream high-resolution video through MMSW. The raster may add image input to a planning request; detailed designs still take provider and execution time.

Lifetime	Meaning
Current Studio scene	IDs and creation order remain valid across turns and client reconnections while the sheet exists.
Clear paper / Reset scene	Live IDs expire. Stale references are rejected, even if new ink occupies the same position.
Studio process restart	A new scene and sheet begin. The PNG archive survives, but is not restored as live ink.
Failed or interrupted drawing	Any actual ink remains and is marked partial. A failed task is not silently recorded as a completed illustration.
Saved drawings	Completed sheets, including enclosures, retain their dark PNG and content metadata in the archive.
Remembered vector recipe	On a nonempty sheet, inherited recipes from earlier pack revisions return through current-paper interpretation before adding geometry.

Public embodiment messages expose only drawing identities, order, status and bounds. They do not broadcast private briefs, inscriptions or requester identities. Task records and the existing archive retain the drawing information needed for the requester and the shared sheet. The live streaming rate remains unchanged.

19.3 Two kinds of addition

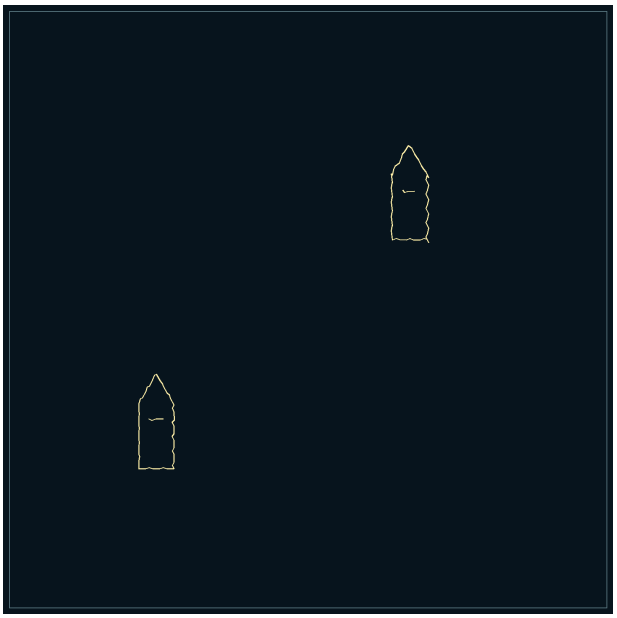
Measured enclosure: `frame_drawing` supports an ellipse, rectangle or rounded balloon around a specified existing drawing. The default outline gap is 8 mm; the allowed range is 5–30 mm. A balloon can point toward an anchor within another drawing's measured bounding box. Its tail stops outside that drawing's ink. The anchor is a spatial interpretation, not a separately verified body-part label.

Other new marks: `annotate` accepts a brief, style, detail and existing drawing references. The design step outputs only the additions in coordinates fixed to the whole sheet. It does not apply the ordinary free-space translation or shrinking. Rays, ornaments and other marks must fit and clear existing ink. This release does not support crossing or editing through existing strokes.

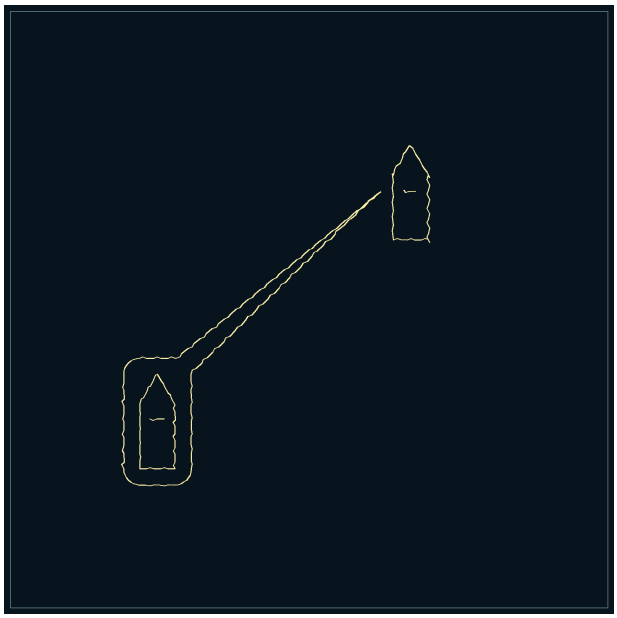
19.4 What is verified

Aster verifies that the new measured pen path follows the planned geometry, stays within the paper and clears scene objects. Before and after checks compare a hash of all previous strokes: the original ink must be identical. New paths reserve 4 mm from prior ink; measured paths must retain at least 1 mm. Ordinary pen tracking keeps the existing 4 mm error limit.

For an enclosure, every measured point belonging to the selected drawing must remain inside the measured closed outline. The original drawing ID and any tail reference are part of the result. A frame adds exactly one stroke, so a new subject cannot be smuggled in as a successful enclosure. Generic provider-designed additions also preserve existing ink, but their semantic content is still reviewed by the model rather than proven by path measurements.



Before: two existing illustrations



After: one added balloon and pointer

Figure 3: Native simulation regression using two simple test illustrations. The original measured strokes are byte-for-byte unchanged. Only the enclosing outline and its tail were added; neither subject was copied. This is a geometry test, not a claim of artistic likeness.

Useful result fields are `enclosure_verified`, `existing_ink_unchanged`, `drawing_id` and `tail_reference`. These are kept in the bounded evidence sent to Celestia. Her report should distinguish the original artwork from the added marks. “The original drawing is now enclosed” is supported by this proof; “the character is an exact likeness” is not.

19.5 Limits and troubleshooting

If a request produces another subject, check that the running adapter reports v59.20260919.91 and Panda revision 20, then inspect whether the task selected `frame_drawing` or an anchored addition. A newly generated free-space sketch is appropriate only when a new drawing was requested. Avoid running an old Studio process with new clients.

This release cannot erase part of a drawing, relocate existing ink, infer arbitrary semantic parts with certainty, or restore an old sheet from a PNG as editable original strokes. Native tests cover separate references, whole-outline containment, preserved ink, page edges, stale IDs, partial ink metadata, archive saving and the complete controlled-provider task transaction. Paid live-provider interpretation and physical Panda hardware are not validated by those tests.

20 Tasks that depend on earlier movements (.73)

20.1 Stack first, then balance the sphere

Aster can now execute this as one complete request:

> Stack the cubes from bottom to top: green, blue, red. Then put the yellow ball on the red cube and check whether it stays balanced.

Previously, planning marked red's future position as unknown and rejected the placement before anything moved. Asking the human to finish the first action was a planning limitation; the arm could attempt this sequence.

Aster now keeps the destination symbolic until the stack is verified. It receives a fresh observation through MMSW, reads the measured object poses, resolves the placement coordinates and executes. This checkpoint stays within the same task; the human does not submit a second request.

The native test passed: green supports blue, blue supports red, and the released yellow sphere stays supported by red with low linear and angular velocities for a continuous second. Landing on the table would fail the balance check.

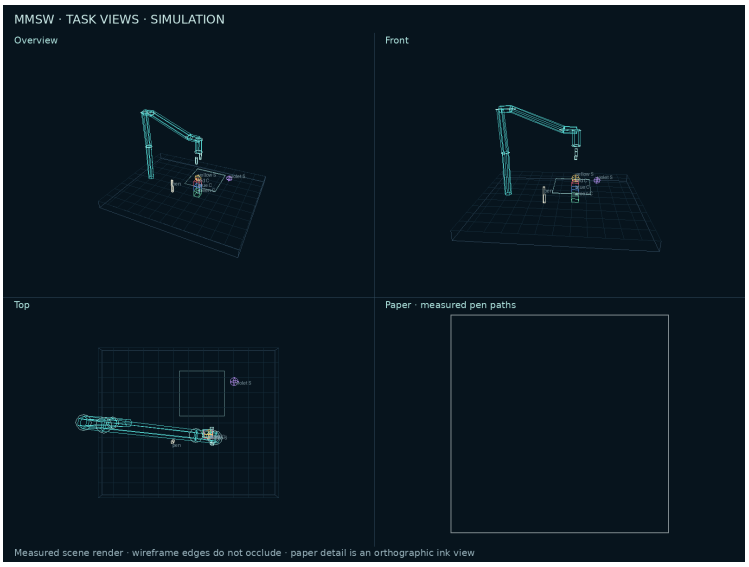


Figure 4: Executed simulation: green–blue–red stack, followed by a yellow sphere balanced on red using its refreshed measured pose. The displayed paper is empty in this test.

20.2 Rebuild an existing stack

If the target stack location is already occupied by the selected cubes, Aster stages them on clear tabletop locations from top to bottom before building the requested order. It uses actual pick-and-place movements, with normal collision checks; it does not teleport cubes or overwrite their poses.

A held two-cube payload must first be released. The tested sequence includes release, top-first staging, rebuilding green–blue–red and balancing the sphere. Blocked staging or a nonselected obstacle can stop the task. Aster reports the measured partial state; an earlier step passing does not prove the complete task succeeded.

20.3 Lift everything by the bottom cube

Use a request such as:

> Stack red, blue and green, with red at the bottom. Grasp only the bottom red cube from the side, then lift the whole stack until the bottom is 15 cm above the table. Keep holding it.

This uses `lift_stack`, distinct from an ordinary single-cube `lift` or horizontal side extraction. The fingers approach and contact the bottom cube; the arm then lifts vertically without withdrawing that cube from under its neighbours. The upper cubes remain dynamic bodies supported by ordinary contact. Only the bottom cube receives the existing simulated grasp constraint.

Evidence	Required outcome
Gripped payload	Only the selected bottom cube is attached and listed in `held_objects`.
Supported payload	Every upper cube appears in `supported_objects` and retains the requested bottom-to-top support chain.
Motion checks	Slow vertical motion uses 60 waypoints, with support and alignment checks after a 50 ms physics interval at each waypoint. This is sampled support evidence, not a continuous force-control measurement.
Alignment and height	Stack lateral drift stays within 12 mm; bottom height is within 6 mm of the requested 5–25 cm table-to-bottom height.
Final hold	All upper cubes keep the expected support and low velocities for a continuous one-second observation window.
Failure	Loss of support, excessive tilt or drift, a collision, or a failed final hold stops the attempt and prevents a whole-stack success claim.

Why the distinction matters. A previous single-cube lift report could verify the red cube's height without proving that blue and green came along. Celestia must use `stack_retained` and the measured `support_chain` before saying the whole stack was lifted. "Red is held" and "the complete stack remains supported" are different facts.

The native test reached the requested 15 cm bottom height with red held, blue supported by red, and green supported by blue. No upper-cube attachment or external force was added. This is a bounded simulator result; no physical Panda driver or hardware stack-carrying procedure is supplied.

20.4 Language and operating setup

The configured provider interprets ordinary wording and likely speech-transcription errors using the visible scene. For example, the scene contains a yellow sphere; it does not advertise a separate yellow bowl. Ambiguous references still require clarification. No phrase-specific rule for "bowl", "coupe" or a particular stacking sentence was added.

Keep the same Guided-only Celestia-led setup and do not submit the same task separately to Aster. The new `place_relative` native action is an internal measured-placement mechanism: ordinary requests still use `place` goals and symbolic destinations. Native placement and collision checks remain active after the fresh observation. See `VALIDATION_v59.73.json` for regression coverage and its limits.

21 No-lift side turns inside a stack (.74)

Panda simulation pack revision 13. A cube between two other cubes can be approached from the side and turned while remaining on its support. Earlier surface turns used a top approach and refused the middle cube because the upper cube blocked it. This section supersedes that earlier top-approach limitation; it retains the no-lift verification limits. The .79 extraction-clearing rule does not substitute an unstacking or pickup for this requested surface turn.

21.1 Give the whole request once

Keep the illustrated Guided setup: Celestia Auto-reply ON, Aster conversational Auto-reply OFF, with both processes and Robotics Studio running. No Expert setting or additional approval switch is needed. For example:

> Stack red, blue and green from bottom to top. Grasp the middle blue cube from the side and turn it a little without lifting it or taking it out of the stack. You decide the angle and direction. The upper cube may topple in this simulation.

Celestia should preserve the blue target, red support, side approach and no-lift requirement throughout delegation. The advertised small-turn default is **15 degrees counterclockwise viewed from above**. If you delegate the angle and direction, she should use that default rather than ask for those parameters again. A missing object or conflicting requirement can still need clarification.

What you request	Behavior
Turn without lifting	Keep contact with the original table or cube support; yaw about the selected cube's centre.
Approach from the side	Close on the side faces without the withdrawal and lift used by an ordinary side pickup.
Keep the upper stack supported	Default `preserve` policy: retain every original upper support link throughout the measured attempt and verify final settling.
The upper objects may fall	Explicit `allow_topple` policy for this simulated attempt: allow their contact dynamics to determine the result and report the measured outcome.
No approach specified	`auto` selects a side grasp when another object rests above the selected cube; otherwise it uses the normal top approach.

Permission is not an outcome. Allowing the upper cube to topple does not mean it fell. In the centred-stack regression it stayed on blue. Celestia must report actual support and motion. The selected blue cube must still meet the no-lift requirement, even when upper-object toppling is permitted.

The arm does not remove the upper cube, grasp a different cube or substitute a lift-and-return motion. A blocked side corridor, unreachable pose, collision or loss of the target's required support stops the attempt. Any measured partial movement and held state must be reported; a failed verification does not mean nothing moved.

21.2 How the turn is measured

Both opposing fingers must contact the selected cube before the existing simulated grasp constraint is attached. Only that cube is attached; upper objects remain ordinary dynamic bodies. A side turn uses slow servo motion and half-degree yaw waypoints, then opens the fingers, withdraws sideways and retreats upward after release. Robot and payload collision checks remain active.

The simulator checks the target's upward support contact at every physics step, including approach, turn, release and retreat. The maximum permitted upward centre excursion is 2 mm and lateral centre drift is 6 mm. These are numerical execution tolerances, not permission to perform a pickup. The final orientation must be within 3 degrees of the target, and the selected cube must settle on its original support. A small downward servo preload helps retain contact; no force-control or hardware-equivalence claim is made.

With `preserve`, upper support links are also checked at every physics step. With `allow_topple`, their losses of contact, displacement, final support and settled state are recorded instead of being mistaken for a preserved stack. A brief loss of contact is distinguished from an observed fall or substantial tilt. No object pose, velocity or friction is overwritten to manufacture success.

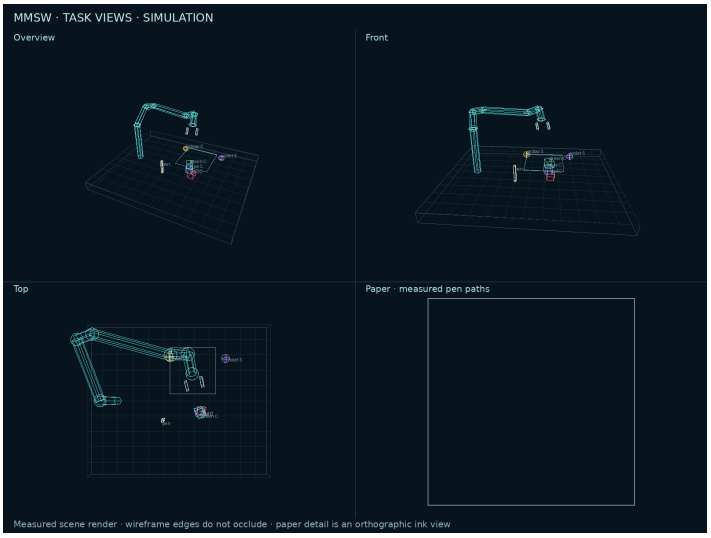


Figure 5: Executed native simulation after a side turn of the middle blue cube. Red remains its support and green remains supported by blue. The permitted-toppling test also reported that green stayed supported.

21.3 Follow-up context and practical limits

The delegation path now retains up to six recent human correction messages alongside the task summary. Context is scoped to the same requester, expires after ten minutes and respects memory reset. Completing or cancelling a task starts a fresh correction history for the next request; permission to topple objects is not a standing permission for unrelated tasks. Aster also retains the bounded clarification history when resolving a direct follow-up. This supplies context to the configured provider without a phrase-specific movement rule.

Native regressions cover 15-degree clockwise and counterclockwise middle-cube turns, automatic side selection, explicit permission for upper toppling, blocked approaches and rejection of incorrect support evidence. The centred-stack runs measured less than 0.42 mm upward excursion and 4.64 mm lateral drift. Controlled-provider tests cover delegation history and a complete stack-then-turn transaction. Live provider interpretation and physical Panda execution are not validated; other poses can still fail their measured checks.

22 Rendered Robotics Studio: materials and lighting

The rendered Observer view shows the Panda's solid links, hand and fingers following measured simulation poses. Painted cubes and spheres, a metallic pen, selectable tabletop finishes, metal table legs, reflections and cast shadows provide a more realistic view. Build .78 upgrades the table materials and edges. The dark-room spotlight and original grey studio remain available.

22.1 Quick setup: a dark room with a spotlight

1. Restart the server, Robotics Studio and clients from the same .78 installation. Open Robotics Studio and force-reload the page once after upgrading.
2. Above the scene, beside **Hide labels**, check **Rendered view**. Uncheck it whenever you want wireframe. Your previous view choice is retained.
3. Find **Lighting & boundary**, in the Observer controls immediately above the scene. Under **Room look**, choose **Dark room · spotlight**.
4. Leave **Room light** at **Off** and **Spotlight** at **100%**. The room is black and the spotlight illuminates the arm, table and objects. These are the defaults when no lighting preference has been saved.
5. Drag either dimmer to adjust its brightness. **0 means off**. **Room lights off** sets only the room dimmer to zero, leaving the spotlight at your chosen level. Set both dimmers to zero to remove the rendered illumination.
6. Check **Show safe-area boundary** for a white strip on the table; uncheck it for an unmarked table. The checkbox works independently of **Show workspace guides** and **Hide labels**.

No Conversation Studio or Expert-mode setting is needed. These controls belong to Robotics Studio's Observer view.

22.2 Choose a more realistic tabletop

Under **Lighting & boundary**, use the new **Table finish** selector. Changes appear immediately in rendered view. No restart, robot command or Expert setting is needed. The choice is remembered on this browser and leaves the room light, spotlight and boundary checkbox at their current values.

Table finish	Appearance
Satin graphite	Default workbench finish: neutral charcoal, fine surface texture and broad, soft highlights.
Brushed aluminium	Silver metal with fine directional brushing and a directional reflective response.
Natural walnut	Warm brown wood with continuous, irregular fine grain and a restrained satin coating.
Classic wood	The earlier striped wood material remains available if you prefer it.

The new finishes use surface-colour and linear microtexture maps with separate height and roughness channels. Metal and wood have different reflection properties. The small bevel around the tabletop catches the light and softens its silhouette. The central working surface stays at the reported height and every bevel vertex stays within the measured table dimensions.

For the dark-room presentation, start with **Satin graphite**, **Room light: Off** and **Spotlight: 100%**. Add a little room light if you want to reveal details in shadow. Use **Brushed aluminium**

for a brighter technical workbench or **Natural walnut** for warmer colour. Switching the lighting preset keeps your table finish; switching the table finish keeps your lighting.

Material changes do not affect friction, contacts, motion limits, the paper, recorded pen paths or saved drawing PNGs. They change this Observer view and its **Save view PNG** export. The texture maps are generated locally once when a finish is first selected, then reused; no texture service or extra MMSW image stream is required.

22.3 Choose the lighting without losing the original rendering

Selecting a **Room look** preset resets the two brightness values to the starting values below. You can then customize either dimmer. Switching between rendered and wireframe does not reset lighting. The room look, table finish, both dimmers and the boundary checkbox are remembered on this browser; another browser can use a different view.

Starting look	Room	Spot	What you see
Dark room / spotlight	0%	100%	Black surroundings with illumination focused on the work table.
Studio / original grey room	100%	0%	The original grey studio background, fill lights and reflections.
Custom dark room	0–100%	0–100%	Keep the black background; add some room light to soften deep shadows if desired.

The room dimmer controls the complete room lighting, including fill lights and environment illumination. The spotlight dimmer is independent. Turning room light off therefore leaves only the spotlight illuminating solid surfaces. Measured pen paths, labels and inspection guides are display overlays and may remain visible with all lights off. Use **Hide labels** and turn off guides for a cleaner presentation.

22.4 Boundary, writing and the other view controls

The white strip follows the safe-zone boundary reported by the simulator. Its centre follows that boundary and its displayed width is 6 mm. In wireframe, the same checkbox shows or hides the existing green outline. If no boundary is reported, none is invented. The strip is a visual marker: **hiding it never changes the safe zone, motion limits, collision checks or permissions**.

Control	Where and what it does
Rendered view	Above the scene, beside Hide labels. Checked: solid surfaces; unchecked: wireframe.
Lighting & boundary	Observer controls above the scene. Room preset, dimmers, table finish and safe-area checkbox. Lighting applies only to the rendered Panda view.
Show safe-area boundary	White tabletop strip in rendered view; green outline in wireframe. Independent of the other guides.
Show workspace guides	Above Lighting & boundary. Toggles the grid, front-direction and sensing-camera guides in rendered view. It does not switch the white strip.
Hide labels	Above the scene. Hides both joint and object labels.
View quality	Economy reduces local pixel resolution and uses 1024-pixel shadow maps; Sharp caps the local pixel ratio at 2 and uses 2048-pixel maps.
Save view PNG	Downloads the current local scene image, including lighting and the boundary if shown. HTML controls and labels are not baked into this PNG.

The paper remains dark charcoal with measured light pen paths. Incoming marks, including unfinished strokes, remain visible above the solid paper. Recorded coordinates are unchanged; objects still occlude ink and clearing the paper removes its marks in both views. Saved drawings remain in their separate measured-paper PNG gallery.

Drag to orbit or scroll to zoom. Slow orbit, Drone, Epic, Slideshow and Drawing sequence also work with either lighting preset. **Expand view** provides more space. If movement feels slow, select **Economy**; the quality setting changes local rendering detail, not arm speed.

22.5 Load, accuracy and compatibility

The browser downloads the bundled Panda visual assets once and normally caches them. The mesh buffer is approximately 5.5 MB before HTTP compression. No online model host or texture service is needed. Subsequent movement uses the existing scene stream. Changing appearance does not send an MMSW command, add a camera subscription, stream extra video or invoke a reasoning model. Lighting and shadows use the viewing computer's GPU; using both light sources can cost more GPU work than a single source.

Object and joint poses still come from simulator feedback. Lighting does not reposition objects or change physics. A stopped or stale connection keeps the existing stale-state indication. This is physically based real-time rendering, not an offline path-traced photograph or a hardware camera feed. Rendering is available for Panda simulation; other arm packs and physical feedback keep their existing views. A missing mesh or WebGL capability retains the fallback behavior.

The sensing camera and automatic multi-view task-result sheet remain separate evidence products. These local lighting preferences do not change those images or saved paper PNGs. The bundled Panda visual assets and their Apache-2.0 license are under `vfxrio/robotics/static/models/panda-rendered/`.

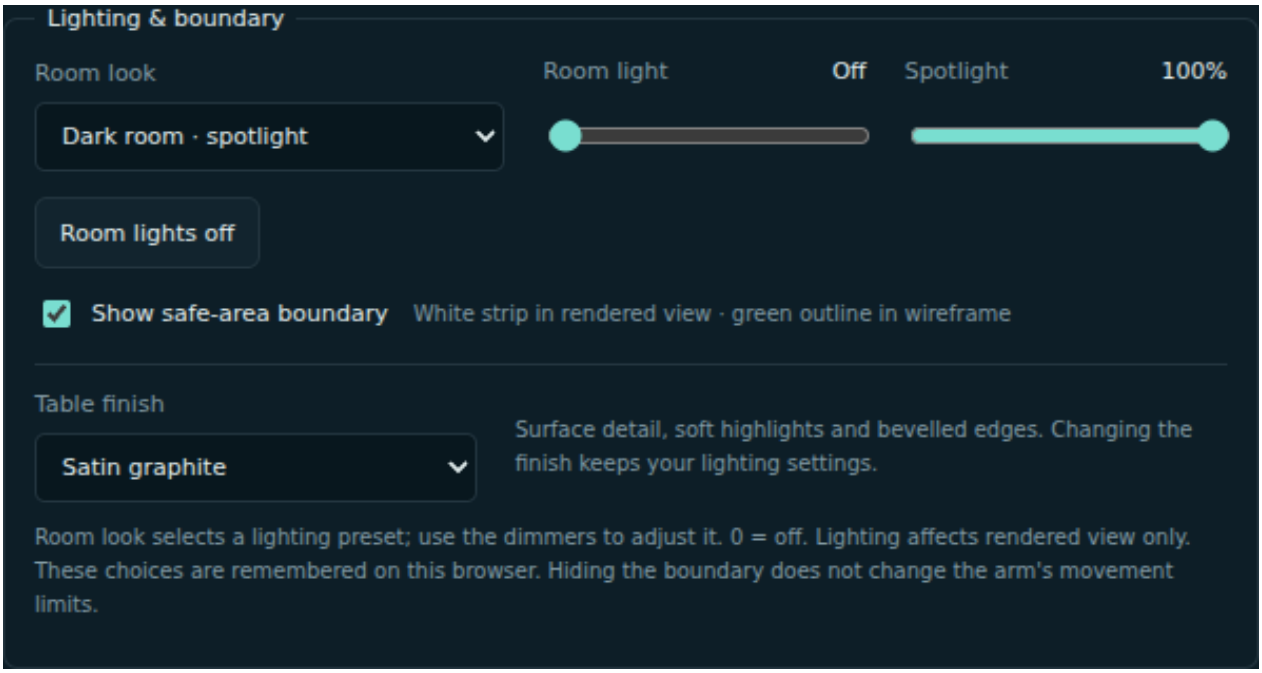


Figure 6: Lighting & boundary in Robotics Studio. Set Room light to Off for the dark-room spotlight setup. No Expert configuration is required.

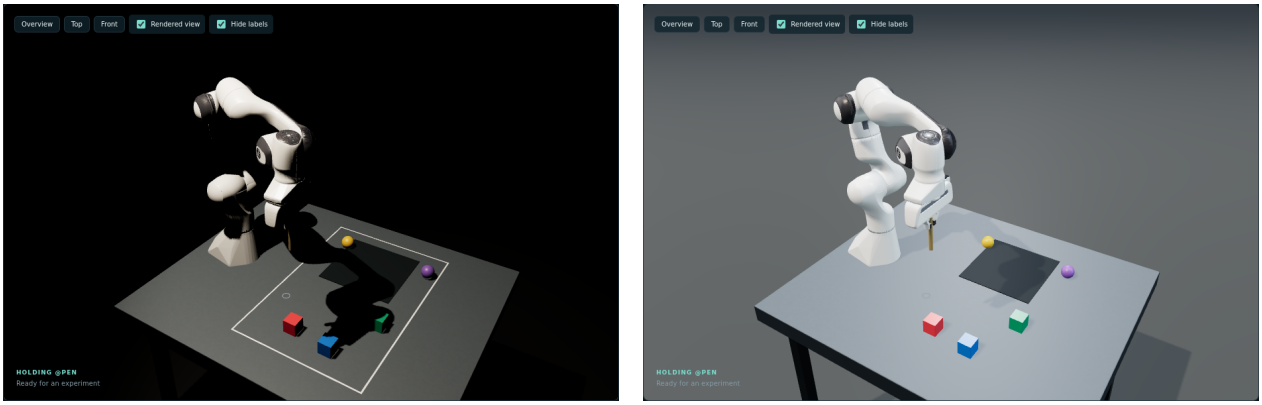


Figure 7: The same measured scene with the .78 satin graphite tabletop: dark-room spotlight with the white boundary (left), and the original studio preset with the boundary hidden (right). Both lighting presets remain available.

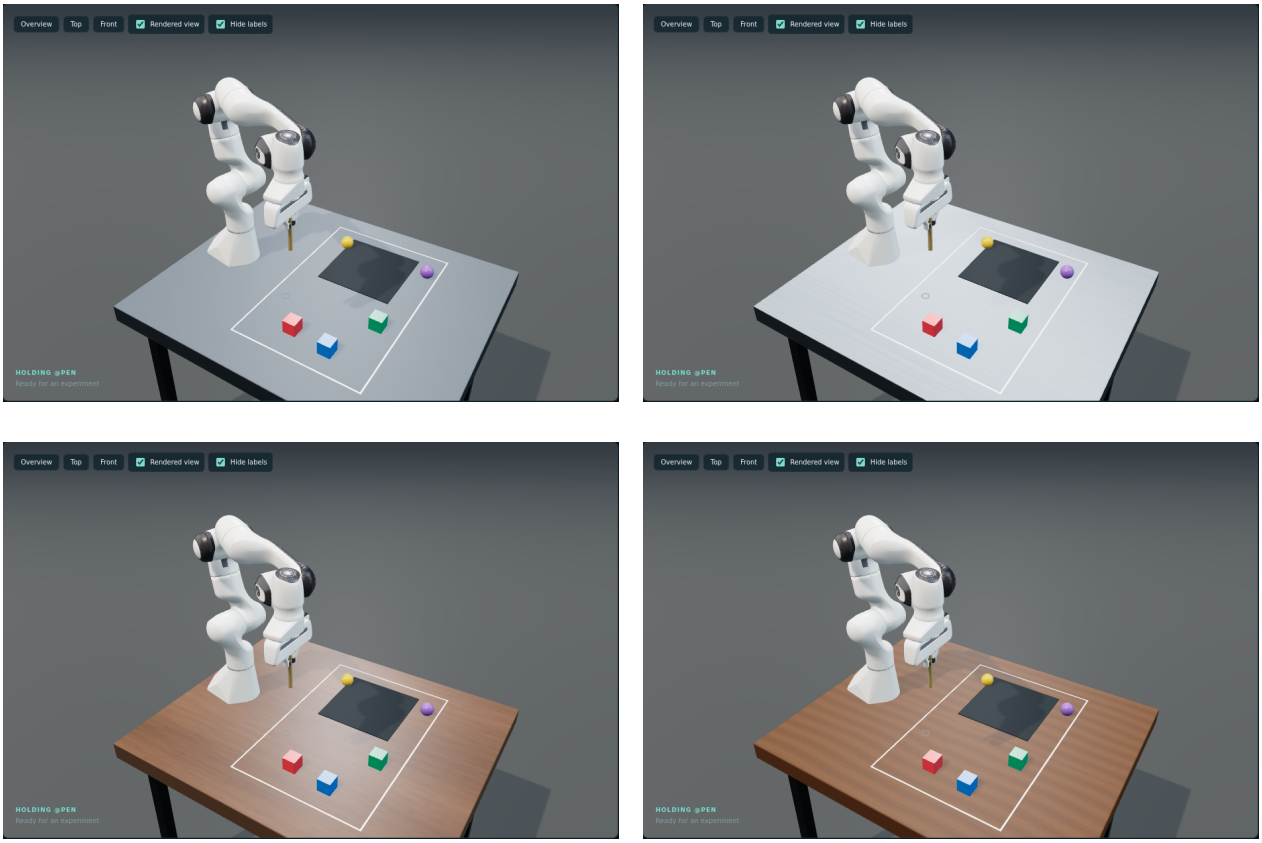


Figure 8: The four selectable tabletop finishes in the actual .78 application under identical studio lighting: satin graphite (top left), brushed aluminium (top right), natural walnut (bottom left) and classic wood (bottom right). The arm, objects, paper and measured scene are the same.

23 Dependent drawing compositions and corrected follow-ups

The planner can now link a new drawing to a later addition in the same task. Previously, an existing drawing could be framed, but a balloon tail could not reliably refer to a speaker that had not yet been drawn. A task-local output reference closes that gap without fabricating an existing drawing ID.

23.1 Examples through Celestia

Keep the usual setup: Celestia Auto-reply ON, Aster Auto-reply OFF, both processes running and bidirectional connections. Conversation Studio is optional; the Guided setup is unchanged.

- “Keep the cat already on the paper. Draw a little ghost near the bottom, then add a balloon from the ghost around the existing cat. Do not draw another cat.”
- “Draw a new helmet in the bottom-left of the paper, then add a line from it pointing toward the nearest left corner of the red cube.”
- “I meant the bottom-left of the paper, not the cube. You can choose the size and composition.”

The second example points *toward* the cube. The pen stays on the paper and stops before its edge or the object’s clearance zone. It does not draw on the cube or bridge empty space. A narrow paper region, crowded ink, unreachable point or blocked corridor can still prevent the requested addition; the result must report that condition, not claim artistic completion.

23.2 How existing and new content remain distinct

Each measured drawing has a current-sheet identity and bounds. A new drawing can declare a short `drawing_key`; a later frame or pointer uses `new:key`. The adapter binds this reference only after the first drawing passes verification. Forward references, duplicate keys, cleared sheets and failed drawings cannot serve as successful sources.

```
{"op":"artwork","brief":"One small ghost only; no balloon or cat",
 "style":"machine plotter","detail":"minimal",
 "drawing_key":"speaker","area_uv":[0.1,0.03,0.85,0.32]}
{"op":"frame_drawing","drawing_id":"<existing measured cat ID>",
 "shape":"balloon","gap_m":0.008,
 "tail":{"drawing_id":"new:speaker","anchor_uv":[0.5,1.0]}}
```

This illustrates the semantic contract; the planner fills the actual current-sheet ID and complete validated artwork fields. The phrase and subject are examples, not special recognizer rules.

`area_uv` is `[left,bottom,right,top]` in normalized paper coordinates. The origin is the paper's lower-left, not the screen corner after orbiting the camera. Automatic fitting stays inside the requested region and avoids occupied paper. When a new drawing will be the source of a pointer, fitting reserves a small margin for the pointer corridor.

A composition containing only trace, frame, pointer and observe actions receives a geometry preflight before new ink begins. Mixed manipulation-and-drawing plans still need live checks at each execution stage because objects can move. Verification compares the actual additions, previous ink digest, reference bindings and layout constraints. It does not certify that a character or artistic style is visually recognizable.

23.3 Clarifications and technical failures

Celestia receives the ordered recent human corrections separately from generated task summaries. The latest clarified meaning replaces a mistaken earlier alternative. If the human selects paper placement after being offered paper versus cube placement, the unselected cube interpretation is not a permanent requirement. Explicit constraints, such as preserving existing ink, remain.

If the human delegates size, angle or composition choices, the reasoning guidance asks the model to choose within the advertised contract. It does not guarantee every provider response or introduce a phrase-specific command list.

An empty or malformed JSON response gets one bounded formatting retry for the same request. No motor action is repeated by this recovery. If both responses are invalid, the result is `planner_response_invalid`, a technical planning failure. It is not reported as a request for a missing object or drawing identity. A failure before delegation explicitly states that no command was sent; pending human corrections remain available for a retry.

23.4 Validation scope

Automated checks exercise real Panda simulation strokes, preservation of earlier ink, a new speaker followed by an existing-subject balloon, a new bottom-left drawing followed by a measured cube pointer, controller execution through the model wrapper, reference forgery rejection, and bounded malformed-response recovery. Provider responses in these integration checks are controlled fixtures; arbitrary live language interpretation and provider latency are not certified. The earlier native stack, side-grasp, middle-cube turn and measured support tests remain part of the regression suite.

24 Spatial awareness and automatic obstacle clearing (.79)

Default: check the workspace, clear a movable blocker, then perform the task. This behaviour is available in `panda.simulation`, pack revision 13. It uses measured simulation geometry and native arm motion. There is no new Conversation Studio setting.

24.1 The simple Celestia workflow

Keep the recommended setup: Celestia Auto-reply **ON**, Aster conversational Auto-reply **OFF**, both processes and Robotics Studio running, and Wake Studio **OFF**. Conversation Studio remains optional. Send one request privately to Celestia, or address her in the configured shared room.

For example:

> Celestia, get the red cube and put it in the centre of the work area.

You do not need to add "clear the path" to each command. If an incidental blue cube blocks the approach or destination, Aster first checks whether it can pick blue up, park it at a clear table position, and still perform the original task. It then executes that clearing motion, verifies it, refreshes the geometry, and continues. Celestia's result includes the cleared object and the original task's measured outcome.

24.2 Choose how obstacles should be treated

Policy	What you can say	What Aster does
`clear` (default)	"Clear any obstruction before doing this."	Preview the motion, move a qualifying incidental blocker, and recheck.
`avoid`	"Leave other objects where they are; stop if the path is blocked."	Preview the motion, leave obstacles in place, and stop on a blocked path.
`allow_contact`	"For this simulation task, don't worry about possible collisions; don't clear the path."	Skip automatic clearing and attempt the requested motion with ordinary contact physics.

An example of the last option is:

> Celestia, get the red cube. For this simulation task, don't worry about possible collisions; don't clear the path.

Celestia must preserve that instruction when delegating. The planner expresses it as `obstacle_policy` on the affected motion goals. The choice applies to this task; the next unrelated request returns to the default. Reusing a saved movement routine does not inherit its earlier contact permission.

Allowing contact does not guarantee completion. Objects remain solid and can move or obstruct the gripper. A servo convergence failure, invalid grasp, unreachable pose or failed task measurement still stops the task and is reported as a failure. Table, self-collision, joint, workspace and outcome checks remain active. This option does not enable or modify physical hardware motion.

24.3 What is checked before clearing

1. Rehearse the action in a separate physics world copied from current measured joint angles, object poses, velocities, payload and drawing state. The live scene is unchanged by this rehearsal.

2. While the gripper is free, look ahead through the remaining steps. This can clear a later placement destination *before* the initial pickup occupies the gripper.
3. Identify the actual obstruction, including arm-link clearance, a carried-object collision, an occupied placement footprint or a blocked object-relative pen path.
4. Search bounded table parking positions outside the drawing paper, other objects and known placement destinations. For cube clearing, test both aligned top-grasp directions when necessary. Rehearse the clearing move and the remaining task.
5. Move the blocker with the arm, measure its released position and settling, check that other objects have not been disturbed, and refresh the original task preview.

The search is bounded to three cleared objects, up to sixteen candidate parking sites per blocker, and two aligned grip directions for a cube. Cancellation remains active during preview, clearing and task execution. A preview is a forecast; live motion still checks contacts and verifies outcomes. An unexpected obstruction during live execution stops the task without blindly replaying partially completed actions.

24.4 Objects and content that stay protected

Task targets, later task objects, relative-placement references, supporting objects and their upper loads normally stay protected from incidental path clearing. The explicit exception is extracting a middle or bottom cube: the default first clears its upper load as described below. The arm will not remove the object being outlined, move a reference used for "left of", or erase existing ink. Ordinary writing and drawing continue to fit available blank paper where possible.

Clearing applies to an independently graspable cube or sphere on the table, or the top object of a stack being prepared for extraction. The pen fixture is not treated as a disposable obstacle. If a drawing task starts with the pen held and needs clearing, the arm can temporarily park the pen at its measured pickup site, clear the blocker and regrasp the pen, provided that whole sequence previews successfully. This is reported in the task evidence.

If the arm already holds a cube or a combined payload, it will not release it implicitly to free the gripper. A task that requires continuous holding keeps that constraint. If there is no verified clearing route or free parking position, Aster names the blocker and stops. Cleared objects stay in their reported parking positions after the task; returning them requires a subsequent instruction.

24.5 Taking a middle or bottom cube from a stack

For a red–blue–green column, bottom to top:

- "Get the blue cube" first parks green on the table, then picks blue.
- "Get the red cube" first parks green, then blue, then picks red.
- "Get blue from the side; it is fine if the stack collapses, do not clear it" keeps the upper load in place and attempts the requested side extraction in simulation. Only blue is grasped; the other cubes remain dynamic and may fall or ride through contact.

The planner interprets the meaning of the request, rather than requiring one exact phrase. Explicit acceptance of collapse or incidental collisions maps to `allow_contact` for the extraction. "Keep everything else unchanged" is different: it does not authorize a fall. With `avoid`, the arm stops before removing a loaded support when it cannot preserve that constraint.

These extraction rules do not replace the other requested physical methods. "Lift the entire stack by its bottom cube" uses `lift_stack` and retains the upper contact chain. A no-lift turn within

a stack keeps its separate `preserve`/`allow_topple` surface-turn policy. A side approach explicitly requested for an ordinary pickup remains a side approach after any default clearing.

24.6 SDK examples and measured reporting

The usual action is unchanged:

```
{"op":"pick","object":"red_cube"}
```

For an explicit simulation contact attempt:

```
{"op":"pick","object":"red_cube",
 "obstacle_policy":"allow_contact"}
```

A structured `task` or `plan` may also carry the field. Semantic goals can carry the same optional field, including an artistic brief; it is preserved when the brief becomes a measured pen trace. Unsupported arm packs reject the option.

The `workspace_preparation` evidence records the policy, whether a preview ran, each relocation's original, target and measured position, whether it passed, any temporary pen handling, and whether the original action started. The final task verification still covers the requested action itself. Contact attempts report measured incidental contacts separately from successful task verification.

Preview and rendering run locally at the adapter. No extra camera stream or large continuous MMSW image traffic is introduced. A clear-path native pickup adds a short physics preview; blocked tasks can take several seconds longer while candidate motions are tested. Progress messages distinguish checking the workspace, clearing and execution. Timing depends on the machine and task complexity.

25 Measured relative placement and side contact (.80)

The reference and requested gap remain part of the task. Build `v59.20260918.81`, Panda simulation pack revision 14, fixes the reported rejection of touching, 5 mm and 2 cm placement. No Conversation Studio setting or special phrasing is needed.

25.1 What to say to Celestia

Keep Celestia Auto-reply **ON**, Aster conversational Auto-reply **OFF**, Wake Studio **OFF**, and the server, Aster and Robotics Studio running. Conversation Studio remains optional. The bidirectional graph connections and Celestia's `--robotics-admin robot-arm` role remain necessary.

Request	Intended and checked result
"Put the green cube touching the side of the red cube."	Aster selects a feasible side, aligns the cube faces, releases green, and verifies actual lateral contact.
"Place the held green cube beside red with a 5 mm gap. Keep red where it is."	The final physical clearance must be within 1 mm of 5 mm; reference motion is monitored throughout the attempted placement.
"Any gap is OK as long as it stays close."	Celestia preserves the active moving object and reference, and can select the advertised 2 cm default. Aster chooses a feasible side and reports the measured gap.
"Put green to the left of red with a 2 cm gap."	The requested side is preserved. Left/right follow world X; front/behind follow world Y. Changing the camera does not change these directions.

"Attached to the side" in this operation means *resting in contact on the table*. It does not glue or weld the objects together, suspend one from a vertical face, or make them a shared payload. Ask separately if you mean one of those outcomes; the current pack does not expose adhesive attachment.

25.2 Why the earlier request was rejected

In .79, a single relative placement could become a plain `place` action with XYZ coordinates before review. That lost the symbolic reference and required final gap verification. The reviewer then asked for `place_relative`, and replanning could produce the same inadequate representation again. Changing from touching to 5 mm or 2 cm did not repair that structural mismatch.

In .80, every relative placement goal remains `place_relative` through review and execution. Its contract includes measurement; the planner does not need to invent a separate verification command. Positive gaps and zero-gap cube contact use the same reference-preserving path. Native execution reads the current poses after previous actions, including rotations or repositioning, instead of relying on predicted coordinates.

25.3 How Aster checks the geometry

1. Read the moving object's pose, the reference pose, object dimensions, grasp frame, and other measured objects. The reference must be settled on the table. Only the selected object may already be held.
2. For "near" or "beside", consider adjacent faces. Check the complete footprint and approach against the workspace and other objects. A named direction remains fixed. Cube faces are

aligned to the reference's measured yaw; the fingers open parallel to the contact interface so they do not occupy the gap.

3. Rehearse each candidate with native motion in a separate simulation world. A failed rehearsal does not move the live scene. Existing obstacle-clearing policy still applies to incidental blockers; the reference is protected. If green is already held, Aster will not silently release it merely to clear a blocked route.
4. Execute a successful candidate with normal gravity, motor, contact and collision checks. For zero gap, approach in bounded increments and stop on lateral contact. Release the moving object and retract the gripper.
5. Verify the final physical separation or contact and one second of stable table support. Monitor the reference pose during motion and settling. Report the measurement and any failure; a nearby XYZ target alone is not sufficient proof.

Measurement	Completion condition
Positive gap	Requested gap 5–150 mm; actual closest-surface separation within 1 mm of the requested value, with positive clearance.
Touching cube sides	Actual lateral contact manifold after release, contact distance within the simulation tolerance (at most 0.5 mm separation or 1 mm penetration), and stable contact for one second.
Reference stationary	Maximum translation no more than 2 mm and rotation no more than 1 degree throughout the attempt. "Stationary" is this measured tolerance, not exact mathematical immobility.
Placement and support	Final object centre within 3 mm of the selected target; both objects settled on the table for one second; gripper empty.

Zero-gap side contact is commissioned for two cubes. A sphere can be a reference for positive-clearance placement, but sphere contact is not silently treated as verified cube face contact. Zero-gap placement beside a loaded reference cube is rejected; clearing that load requires its own explicit task. These measurements describe the digital twin; no physical Panda driver is added by this release.

25.4 Evidence and capability refresh

A successful record includes `relative_placement`: requested gap, measured gap, resolved side, contact/clearance result, maximum reference drift and rotation, release, and stable support. Celestia receives the compact result fields and can distinguish "touching verified" from "measured gap 5.00 mm". Failure to meet the requested measurement remains a failed task; it is not converted to a successful generic placement.

Restart the updated server, Robotics Studio and clients. Confirm v59.20260919.91 and `panda.simulation` revision 20 in the runtime and pack status. The hidden capability exchange refreshes Celestia's SDK description when the connected Aster advertises the new manifest. No wake rule, pasted command list or manual lexicon entry is required. Verified routines from compatible revisions remain discoverable and are revalidated; reference-relative recipes are resolved again against the current scene.

26 Language, Persona and an embodied goodbye

Build .81 / Panda simulation pack revision 15 / documentation revision 20. Celestia can use her authorized, connected simulated arm to express a genuine farewell. The robot still executes a bounded command and measures its result; ordinary dialogue cannot manufacture a motion receipt.

26.1 Use the same simple Guided setup

Keep the server, Robotics Studio, Aster, Celestia and Matteo running. Start Celestia with the robotics role:

```
sh scripts/run-celestia-v59.sh 127.0.0.1 --robotics-admin robot-arm
```

Connect Matteo ↔ Celestia and Celestia ↔ Aster in both directions. The role and graph connections authorize this integration; merely viewing a camera does not.

Setting	Celestia	Aster	Matteo
Auto-reply	ON	OFF	OFF
Wake Studio	OFF	OFF	Not needed
Guided Session role, if Studio is enabled	Lead / facilitator	Excluded	Human guide
Guided Open conversation	0: Direct / assigned only	Not needed	Not applicable

Private requests need no Conversation Studio. For the shared room, use Guided: **Collaborative work**, **Collaborative session**, **Keep current Entity settings**, **Fixed primary host**, and choose **Celestia Borealis** explicitly. Keep the temporary facilitator at None and click **Configure room**. Appendix A shows the controls. Aster's Excluded role affects Studio speaking turns; delegated tasks still execute while its conversational Auto-reply is OFF.

26.2 What happens when you say goodbye

1. Tell Celestia, for example, "Can you hold the green cube?" Wait for the measured grasp result.
2. Say "Goodbye, see you later." The configured reasoning provider interprets this as your farewell to Celestia. It is not a literal keyword trigger.
3. With a current revision 15 capability offer, Celestia sends one command for a safe empty-hand farewell. She gives a brief natural goodbye in the active conversation language while that command runs.
4. Aster checks the actual held state. A single cube or sphere is placed in clear table space outside the paper. A held pen returns to its current learned parking fixture. An upright, aligned pair already held through the combined-grasp skill is lowered and released together as a supported column.
5. Each released member must have measured support and remain settled for one second. Only then may the empty gripper perform the bounded wave. If already empty, it can wave directly.
6. The terminal record must verify both parking and the gesture before Celestia reports completion. Aster's matching result narration cannot start another automatic command.

Safe placement, not a drop from height. If a clear reachable parking path or stable support cannot be verified, the wave stops. Cancellation, graph routing, controller ownership, joint limits and collision checks remain active. The final reply describes measured state, including any partial motion.

The farewell speech and arm command run asynchronously. This supports conversational overlap; it does not promise frame-accurate synchronization between speech and finger motion. If speech generation fails, execution evidence is still retained and the terminal result can be reported.

Human turn	Intended interpretation
“Goodbye, see you later.”	One social farewell with supported payload parking, then a wave.
“Goodbye, but don’t move the arm.”	Conversational goodbye; no social motion.
“What does a goodbye wave mean?”	A question, not a gesture command.
“Translate ‘goodbye’ into Italian.”	Translation request; no farewell motion.
“Keep holding the cube and wave.”	Explicit carry-object gesture. Default retain policy remains available.
“Park the pen, then wave goodbye.”	Complete explicit task; each action must verify.

The social route requires a human turn directed to Celestia and a current supported pack. Agent chatter, duplicate results, quoted instructions, negations and third-party farewells must not initiate it. An unavailable or expired capability offer does not grant motion permission. Unfamiliar wording is handled by the configured reasoning provider, not a growing list of language-specific trigger phrases.

26.3 Language continuity and active character

The original human request and bounded recent human turns now accompany the result reply. Aster’s English task paraphrase or technical execution record is evidence, not the human’s language choice. The latest human turn is also the query used by the profile-skill selection and response finalization stages.

In an English conversation, “Can you hold the green cube?” should receive an English response. A name such as Matteo, Italian profile prose or a previous robot message does not select Italian. If you actually switch conversation language, or say “Please reply in Italian from now on”, subsequent dialogue should follow that change. A short “yes”, an object name or a quoted foreign word uses the surrounding human context.

Persona and State of Mind remain your client settings. Use Celestia’s own Persona Studio and Mind Studio to load the intended profile, skills and active state. This build does not silently turn disabled features on. The active human-facing Persona branch and State of Mind shape the goodbye and the measured task response; the same style, evidence and conformance stages used by ordinary replies finish them. The human’s actual request is retained through those stages. Post-delivery Persona/Mind observation is now included on delegated replies too.

For an embodied social command, Celestia supplies a bounded expression profile from her active Persona/Mind. It can change gesture amplitude, pace and cycle count within the implemented limits. Ordinary direct commands to Aster still use Aster’s own profile. Personality never changes an execution failure into success or turns simulator evidence into a hardware claim. The startup field `mind_state_machine=off` concerns automatic state evolution; it alone is not proof that a selected Persona or manually active Mind state was absent.

26.4 Fix for the oversized hidden capability message

The reported warning `Robotics capabilities packet exceeds 96 KB` means the hidden command catalog could not be sent. It can leave Celestia without the current offer even though camera frames and ordinary tasks still work.

Revision .81 removes repeated JSON Schema fragments using local schema references. Every command and schema constraint remains available; the protocol limit is unchanged. The full resulting packet is checked against the 96,000-byte limit, and expanding the references must reproduce the original planning schema exactly. Ordinary conversational prompts receive the command descriptions once, without the large machine-validation schema or a second copy of the same SDK.

After updating, restart all five processes from the same .91 directory. Confirm v59.20260919.91 and `panda.simulation revision 20` in startup or task evidence. Connected clients renew the hidden exchange automatically; no Wake rule or pasted command list is needed. If the old warning remains, first check for a still-running .79/.80 Studio process and confirm the path printed under **Arm executor**. Do not mix launch terminals from different extracted builds.

26.5 Verification scope

The regression suite checks full capability serialization and schema equivalence; the human-language context through reply generation and finishing; active Persona/Mind prompt content; duplicate-result suppression; safe parking and empty-hand wave evidence; held cube, sphere, pen and combined-pair cases; and failure before a wave when parking is blocked. Provider responses are controlled fixtures for deterministic tests. These tests do not establish how every live model will interpret every farewell or language change, and they do not commission physical Panda hardware.

27 A shared landing surface across two cubes

Build .82 / Panda simulation pack revision 16 / documentation revision 20. The arm can place one cube across the top faces of two adjacent cubes. Both lower cubes remain in place; the upper cube is released and supported by physical contact on both. This covers the green-on-red-and-blue request and the blue-on-green-and-red example in the supplied logs. Colours do not select different motion rules.

27.1 Ask for the shared surface directly

Use the same private or Guided shared-room setup from Appendix A: Celestia Auto-reply ON, Aster Auto-reply OFF and Wake Studio OFF. No Expert setting or extra permission switch is needed.

> Celestia, place the green cube centred on top of both the red and blue cubes, spanning their gap. Keep both lower cubes where they are.

If the supports first need arranging, include that in the same request:

> Place blue beside red with a 10 mm gap, then place green centred across the top of both.

The second action refreshes the measured geometry after the first. Do not ask for a table position "between" the cubes when you mean above them. Celestia should preserve the vertical relationship from your earlier request and should not repeatedly ask you to select only one support. A bare "yes" to two incompatible alternatives is not a reliable selection.

27.2 Why a small gap can work

With two level 50 mm cubes separated by a 10 mm gap, a centred 50 mm cube spans the opening. In ideal aligned geometry, 20 mm of its width rests on each lower cube and 10 mm spans the gap. That is approximately 40 percent of its footprint on each support and 20 percent unsupported. The measured fractions can differ slightly after settling.

In everyday language, "half on each" usually describes this centred bridge. It does not require each support to cover exactly 50 percent of the upper cube's underside. Literal 50/50 full-area support would require no gap. This distinction avoids the unnecessary rejection shown in the supplied log.

Check	Required outcome
Base geometry	Both supporting cubes are flat and settled on the table; top heights differ by at most 2 mm.
Shared footprint	At least 10 percent of the payload footprint overlaps each named support; its projected centre of mass lies at least 2 mm inside the combined support polygon.
After release	Positive upward contact on both named supports and low velocity persist for one continuous second.
Placement accuracy	The released cube centre is within 3 mm of the measured centred target.
Stationary bases	Neither support moves more than 2 mm or rotates more than 1 degree during execution.
Empty gripper	No object is attached after release; the upper cube is not welded or constrained to the bases.

27.3 What is verified and what can fail

The controller computes the landing location from the current top faces, including their yaw. It uses native pickup, servo transport, lowering and gravity settling. The supports remain dynamic bodies with normal collisions. A contact on only one cube does not pass. Landing on the table does not pass. Reference drift, an excessive gap, a tilted or loaded base, an unreachable path, or an occupied corridor can stop the task. The controller does not silently move the bases, choose one of them instead, or disable collision checks.

The execution record includes the two support IDs, measured overlap fractions, unsupported fraction, centre-of-mass margin, upward contact counts and normal forces, stable time and maximum base drift. The scene's `motion.supports` lists all measured resting supports; the older singular `support` field remains for compatibility. Workspace preparation follows both support links, so later work must not mistake one bridge base for an unrelated obstacle.

The planning interface accepts:

```
{"op":"place","object":"green_cube",
 "destination":{"across":["red_cube","blue_cube"]}}
```

It compiles to `place_across` with symbolic support IDs, never guessed XYZ. Connected Celestia receives the new capability through the normal hidden exchange. Restart all processes from build v59.20260919.91 and confirm `panda.simulation revision 20` if the old single-support clarification still appears.

Scope: one upright cube across two flat tabletop cube supports in the Panda simulator. This does not commission a physical Panda, certify arbitrary balancing structures, or promise that every support arrangement is reachable. Language interpretation remains the configured provider's responsibility; the native controller verifies the physical outcome.

28 Fine pulling, contact pushes and hand approach

Build .84 / Panda simulation pack revision 18 / documentation revision 22. These controls retain the fine-control work prepared for .83 and ship with the image-pose update. They use measured geometry and native motion. The same private Celestia setup and illustrated Guided shared-room setup apply; no Expert control is needed.

28.1 Pull without lifting, then keep holding

> Celestia, grasp red from the side and pull it 3 cm toward the robot without lifting. Keep holding it.

`pull` performs a side grasp, a horizontal translation of 5 mm to 12 cm, and continued holding. The default distance is 3 cm. `toward_robot` uses the measured robot base. `auto` selects a clear reachable direction. A two-number world XY vector gives a specific direction. A human-relative phrase such as "toward me" needs a known human location; it must not silently mean "toward the robot."

The selected cube must already rest on the table or a measured support. The controller monitors support contact throughout the grasp and pull, with at most 2 mm vertical drift. Measured travel and cross-track error must be within 3 mm; yaw drift must remain within 3 degrees. No lift-and-return substitution passes these checks. A second small pull can retain the existing grasp.

28.2 When a load or neighbouring cube is in the way

With the default clearing policy, pulling a bottom or middle cube first parks the upper load, top-first. This also covers a bridge: if green rests across touching red and blue bases, pulling red first parks green and protects blue. The full hand and payload path is checked in a separate simulated world before execution; proximity alone does not require moving a neighbouring base.

> Pull red 3 cm toward the robot. Clear the cube above it first and keep blue where it is.

For an explicit simulation collapse experiment:

> Pull the bottom cube from the side without clearing the stack. It is OK if the upper cubes fall; keep the selected cube on the table and keep holding it.

The current task uses `allow_contact`; upper cubes remain dynamic bodies. Joint, self-collision, table and workspace limits still apply. This preference does not become a permanent setting or authorize physical hardware contact. If you want the whole stack carried by its bottom cube, say "lift the entire stack by its bottom cube"; that uses the separate `lift_stack` capability and verifies the complete support chain.

28.3 Close the fingers and approach without touching

> Close the gripper, then move near the yellow sphere without touching it.

Independent `gripper` width zero closes an empty hand. `approach` then preserves the opening and moves the empty hand near a measured object. Default clearance is 3 cm, with a supported range of 1–10 cm. Select `auto`, `above`, `left`, `right`, `front` or `behind`. Left/right is world X-/X+; front/behind is Y-/Y+. This is distance from the actual hand geometry, not a guessed TCP offset.

The task fails if the hand contacts the target or moves it by more than 2 mm. The final measured clearance must be within 8 mm above the requested value, and finger opening must remain within 2 mm. A held payload must first be placed; it is not silently released by `approach`.

28.4 Small contact pushes, including spheres

Push the yellow sphere off the red cube with the fingertips and let it settle below.

`push` now supports spheres as well as cubes. Spheres may roll under actual contact physics. Explicit `slide` supports travel from 5 mm; the short-slide verification checks the requested travel instead of accepting a large displacement. `off_support` must actually leave its measured support, fall, and settle below. It is not a moving throw or a grasp-and-drop. Unspecified pushes retain the existing smaller, support-aware `auto` behavior.

28.5 Measured records and persistent routines

Pull records include requested/measured travel, direction, support, continuous contact samples, vertical drift, grasped objects and neighbour displacement. Approach records include final and minimum clearance, target drift and finger opening. Ordinary result replies retain the relevant measured fields. Stored routines from known earlier compatible Panda revisions remain visible within their original requester and conversation scope and are revalidated against the current pack and scene. Unknown future revisions are not assumed compatible.

Simulation scope: The side grasp remains contact-qualified and constraint-assisted. These tests do not establish real force closure, human hand dexterity, or hardware commissioning. Reachability and measured outcomes can still stop a task.

29 Copy an arm direction from an image description

Build .84 / Panda simulation pack revision 18 / documentation revision 22. Celestia can use a visual description she just gave the same human when that human asks the connected arm to imitate it. The description is carried into robotics dispatch with its source and recent conversation. It no longer disappears simply because the next request enters the robot-task path.

29.1 Simple Guided setup: no new switch

Keep Aster and Robotics Studio running. Start Celestia with the robotics role:

```
sh scripts/run-celestia-v59.sh 127.0.0.1 --robotics-admin robot-arm
```

Connect Matteo and Celestia bidirectionally, and Celestia and Aster bidirectionally. Use the established settings:

Setting	Celestia	Aster
Auto-reply	ON	OFF
Wake Studio	OFF	OFF
Private requests	Send requests here	Executes delegated tasks
Conversation Studio	Optional	No automatic speaking floor

For a shared room, use Guided: Collaborative work, Collaborative session, Fixed primary host, Celestia. Choose Celestia's Lead/facilitator role and Aster's Excluded conversational role; keep the robot client running. Apply the configuration and check Celestia is available. The illustrated controls and connection checks remain in Appendix A. No Expert setting or Wake rule is required for image imitation.

29.2 The two-turn interaction

1. Send the image or its accessible image URL to Celestia: "What do you see here?"
2. Wait for her description, for example: "A person has one arm raised and the hand points upward."
3. Ask: "Can you do that with your robotic arm?" or "Mimic the arm direction you just described."
4. Celestia should delegate the upward directional pose from that description. Aster checks the available path, poses the hand upward and keeps the pose. Celestia reports the measured result.

Describing the image alone does not move the arm. A short "do that" can refer to the recent description; it does not replay completed robot tasks. If the image contains several conflicting arm poses, one question identifying which pose is appropriate.

You can also ask in one turn:

> Copy the arm direction in this image with your robotic arm: [image URL]. Put down anything you are holding safely first, then keep the pose.

The existing multimedia reader supplies current visual notes before dispatch. Normal follow-ups reuse scoped notes and do not fetch the image again. The default `perception=auto` setting supports this route. If image reading is disabled or fails, an already delivered description or a human description can still supply direction; otherwise Celestia should explain the missing visual fact rather than claim to see it.

29.3 What the Panda reproduces

`point_pose` orients the hand's longitudinal +Z axis toward the requested direction and leaves its motors holding that pose. Named directions are up/down (world Z+/Z-), left/right (X-/X+) and front/behind (Y-/Y+). A finite nonzero XYZ direction vector supports oblique poses. The controller selects its own bounded reachable hand position and wrist roll; it does not infer an XYZ position from pixels.

An upward-pointing person supplies enough evidence for an upward hand direction. The Panda has a parallel gripper, so it cannot reproduce an isolated human index finger, a full human skeleton or the exact elbow shape in a photograph. This is a measured directional approximation. Image-plane left/right is not evidence of hidden depth; a depth-specific request may need one clarification.

```
{"op":"point_pose","direction":"up","payload_policy":"park"}
{"op":"point_pose","direction":[0.5,0.2,1.0],
 "payload_policy":"retain"}
```

The first example gently parks and verifies any held payload before posing. The second explicitly retains it, provided its full path is feasible. Default `park` is supported placement, never an airborne drop. The arm does not automatically return home after the pose. Say "home" or give the next task when ready.

29.4 What Aster measures

The controller searches bounded IK candidates, checks joint limits and the complete joint path, rehearses motor convergence, then executes through joint motors. It measures the resulting hand axis, tool position, joints and held state after a 0.5 second hold. Direction error must be at most 3 degrees and tool-position error at most 8 mm. Safe parking uses the existing supported-placement verification. Explicit retention uses the existing payload-retention checks.

The record identifies the approximation as `hand_direction`; successful motion is not proof of visual likeness, human anatomy reconstruction or physical Panda validation. A blocked or unreachable pose reports failure or partial progress from actual evidence. It must not claim that nothing moved if payload parking already occurred.

29.5 Memory boundaries and troubleshooting

Visual context is bounded to the same human, private/shared audience, memory epoch and a 30-minute window. A newly supplied image replaces the active reference; an unavailable new image must not borrow an older picture's description. Resetting active memory or restarting the client ends this temporary visual context. This is separate from persistent, verified movement routines.

If Celestia asks for a picture she just described, confirm all processes run build v59.20260919.91 and that the task evidence advertises `panda.simulation revision 20`. The hidden capability exchange refreshes on connection. The complete catalog stays under 96 KB by sharing repeated schemas and keeping repeated rule blocks once in their named SDK sections. No commands are truncated. In a new audience or after a reset, resend the picture or describe the direction once.

Validation limits: The native directions, parking, retention, cancellation, context isolation and dispatcher/controller handoffs were tested locally. Provider-payload tests use controlled responses; they do not guarantee every live model interpretation or external URL will succeed. The photograph's description is visual evidence only; simulator state remains motion evidence.

30 Optional expressive feedback from Celestia

Build .85 / documentation revision 23. This option changes how the Entity reports a delegated arm result to the human. It uses the Entity's enabled character and State of Mind to interpret and express the experience while keeping Aster's measured outcome authoritative. It is **OFF by default**.

30.1 Choose the feedback style

Setting	OFF: current behavior	ON: expressive feedback
Response style	Existing robotics response pipeline and configured behavior	More explicit use of enabled Persona, relevant skills, Personality and current Mind in the response
Result-image link	Appended when a valid frame is available and upload succeeds	Omitted from the Entity's human-facing reply; no result-image upload is initiated for this reply
Measured facts	Authoritative	Equally authoritative; failures and uncertainty remain visible
Task evidence	Available in Robotics Studio	Remains available in Robotics Studio; the reply may still use the image as internal evidence
Disabled features	Stay disabled	Stay disabled; this switch does not load a Persona or enable Mind

OFF restores the pre-.85 behavior; it does not globally disable Persona or Mind. Those features keep their own settings. This distinction lets an operator keep the existing precision-oriented feedback and image delivery whenever wanted. ON adds no extra mandatory model call: it changes the guidance in the existing generation and finishing path. Enabled conformance or guardrail stages keep their normal behavior and cost.

30.2 Where to click: no Expert mode required

1. Open **Celestia's HTML5 client**, using the address printed by `run-celestia-v59.sh`. The bundled configuration uses `http://127.0.0.1:56440/`; use the printed address if you changed the port.
2. If the sidebar is hidden, click the menu button at the top left. Expand **Robots · sensing camera**.
3. Below the camera and owned-components information, find the checkbox **Expressive robotics feedback**. Its caption identifies the local replying Entity. Check it for ON; uncheck it for OFF. The selected robot does not change.
4. Use Celestia's **Persona Studio** to load/edit her character and relevant skills. Open **Mind Studio** from her sidebar and enable/configure Personality or State of Mind if desired. The feedback switch respects those independent settings.
5. Ask Matteo's next question through Celestia as usual. You can also click **Config** in Celestia's client and inspect `ExpressiveRoboticsFeedback` to confirm the effective setting.

Change the switch in **Celestia's client** when Celestia relays Aster's result. It is a per-Entity presentation preference, not a robot motor control. The existing **Persona task replies** and **State of Mind** switches in Aster's own client control Aster and are separate from this option.

30.3 Commands and startup preference

In Celestia's curses input or HTML5 chat input, use:

```
:robotics_feedback=on
:robotics_feedback=off
:robotics_feedback
```

The last command displays the current state. These commands are local controls; typing them in Matteo's client changes Matteo's setting, not Celestia's. The checkbox and commands apply until that client restarts. To start Celestia with expressive feedback enabled, use:

```
sh scripts/run-celestia-v59.sh 127.0.0.1 \
  --robotics-admin robot-arm --robotics-feedback on
```

Use `--robotics-feedback off` for the current behavior. For a reusable launch configuration, add `"robotics_feedback": "on"` to Celestia's JSON configuration. An explicit CLI argument overrides the configuration value. The supplied default remains OFF; changing Persona profiles does not change this preference.

The style is captured when a feedback reply starts preparing. Switching OFF affects the next reply prepared. If you switch ON while a reply is already being prepared or its image upload is running, its image link is still removed before delivery. An upload already in progress cannot be undone. Earlier chat messages and links remain in the transcript.

30.4 Character and perception without changing the result

Enabled Persona skills and the human-facing style branch are resolved using the human's actual request. Enabled Mind can affect what Celestia attends to, how she appraises the outcome, her emotional tone and her perspective on the measured scene. For example, a curious character might notice an interesting arrangement; a subdued state can express satisfaction quietly. The reply must distinguish such appraisal from a fact about the world.

The following are **illustrative wording**, not fixed responses or live-model test outputs. With a verified simulated green-cube grasp and active warm Persona, Celestia could say: "There we are—I've got the green cube. The simulated grasp is verified, and I'm keeping it held." With a failed drawing and a held pen, she could say: "That line didn't land as intended. The drawing did not pass verification, and I'm still holding the pen." ON attaches no result-image URL to either response. Requested measurements should still be included.

Character cannot turn a failed, cancelled or unverified attempt into success. Pen-path verification does not prove visual likeness; holding the bottom cube does not prove the rest of a stack was retained. Simulation remains simulation. The existing finishing, conformance and configured guardrail path continues to receive the measured record. If generation or verification falls back, the client uses factual record-based wording, which may have less character. Image-link suppression still applies to that fallback.

30.5 Recommended task setup and scope

Keep **Celestia Auto-reply ON, Aster Auto-reply OFF and Wake OFF** for both. Maintain Matteo ↔ Celestia ↔ Aster connections and keep Aster and Robotics Studio running. Private delegation needs no Conversation Studio. In a shared Guided session, keep Celestia as fixed primary host and Aster excluded from ordinary speaking turns, as in Appendix A.

The result remains correlated with the original request and is returned once to that requester. This preference does not start another movement, change arm limits, alter the task record, mute sensing-camera notices, or hide Aster's independently addressed messages. Keep Aster's conversational Auto-reply OFF for the Celestia-led workflow. Use `:system_messages=off` in Matteo's curses client separately if camera/system notices are distracting.

30.6 Validation limits

The release includes automated checks for the default, both toggle directions, CLI/config precedence, live UI state, active and disabled profiles, human-language context, guardrail fallbacks, private correlated delivery and image suppression after generation or an in-flight toggle. Provider replies are controlled fixtures; the exact literary quality of a live Persona/Mind response depends on the selected model and profile. This change does not modify or newly commission physical arm motion.

31 Reliable drawing review, feedback and movement transitions

Build v59.20260919.91 addresses the reported .84/.85 failures while retaining expressive Entity feedback, the illustrated Guided setup and all rendering controls. Restart all five processes from this extracted build so the server, clients and Studio use the same protocol implementation. Panda advertises pack revision 19.

31.1 Drawing above existing ink

You can say: “Draw a tetrahedron above the circle, preserving the existing ink.” The planner selects the existing drawing and a clear paper region. The adapter fits the new geometry within that region, preserving its proportions and checking the existing ink and object envelopes. It checks the layout again before motion. There is no tetrahedron-specific drawing template or language rule.

Earlier review messages compared world Y positions such as 0.26 metres with paper coordinates such as $v = 0.7045$. Those values use different frames and cannot be compared directly. The review now receives labelled world bounds and normalized paper bounds, including the fitted layout when it can be computed before motion. Normalized paper u runs right; v runs up; both range from zero to one. The requested `area_uv` constrains the final fitted drawing. An anchored addition retains its fixed position instead of being fitted elsewhere.

If preceding actions move objects, the final layout is deferred until fresh feedback is available. This is part of the same task; it does not require a new human instruction. An actual lack of free space or a blocked pen path can still prevent the drawing. A plan-review failure before motion and a collision during execution are different outcomes and should be reported as such.

31.2 Pose imitation and return to rest

An observed arm direction is a directional approximation, not an exact human-body reconstruction. For example, the vectors $(-1, 0, 1)$ and $(-1/\sqrt{2}, 0, 1/\sqrt{2})$ point in the same direction. Normalizing a direction must not cause a request for the literal, unnormalized vector.

Try a sequence such as “Point up and left”, “Return to rest”, then “Get the pen”. The simulator uses a checked joint path when it must reorient a raised empty hand for a tabletop task. When a hand is near the table, it lifts with its current orientation before changing orientation. Joint limits, table contact, object contact and measured endpoint checks remain active.

31.3 Clearing a stack and picking two cubes

For a blue–red–green stack, a shared side grasp of blue and red selects two payloads. With the default clearing policy, green is an external upper load and can be parked first through verified native movements. The selected pair stays the requested pair. The adapter rehearses alternate sides and wrist tilts, including the forearm, fingers and initial lift, before choosing a live approach. This can avoid clearing an object that obstructs only one possible approach. With `avoid`, the upper load stays in place and the blocked attempt stops before live motion. Explicit acceptance of incidental collisions applies only to that simulation task and never disables table or joint limits.

When building or rebuilding a stack, each completed cube becomes a measured support for the next cube. The next placement uses that support’s actual height, including small settling offsets, rather than an idealized column height. This avoids mistaking the intended supporting cube for an unrelated obstacle. For a no-lift middle-cube turn, the final side insertion is slower and finer; continuous support checks still stop a movement that lifts or disturbs the stack.

Policy	Meaning for this task
`clear`	Default. Preview and move incidental blockers when a valid grasp, parking space and path exist. Clear external upper loads before extraction.
`avoid`	Preserve obstacles; stop when the requested path is blocked.
`allow_contact`	Explicit simulation contact experiment. Skip incidental clearing; retain contact physics, table, self-collision and joint checks.

An accepted collapse does not guarantee a successful grasp or a collapse. Only measured support-chain evidence establishes that an entire stack was carried. Holding one cube or a pair is not evidence that another cube remained on top.

31.4 Feedback that reaches the conversation and Mind Studio

Long learned drawing recipes are compacted for transport while the full recipe remains in the adapter ledger. The server now formats both the ordinary recipe string and its compact record, so a completed drawing does not get stuck in the repeated "Feedback will retry" error caused by concatenating a dictionary. Pack revision 19 also retrieves older verified recipes within the same owner and conversation scope, then revalidates them against current capabilities and state.

Celestia's delivered result now uses the current Studio telemetry module and function signatures. This repairs the missing-module error after robotic replies. A telemetry failure does not prevent the separate background runtime assessment. The assessment observes the delivered reply; it does not issue another movement.

Signed rotation measurements remain prominent in the concise evidence used for feedback. Around world Z, a negative angle is clockwise when viewed from above. Persona and Mind expression must retain that measured direction and viewpoint.

The **Expressive robotics feedback** checkbox remains optional, OFF by default, in **Celestia's HTML5 client → Robots · sensing camera**. ON uses the enabled Persona/Mind style and suppresses outgoing result image links; OFF retains the existing feedback presentation. See Section 30 for commands and startup settings. Guided setup still needs no Expert controls.

31.5 Interpreting the logs

Successful HTTP 200 asset requests and a WebSocket 101 connection are ordinary Studio traffic. They are not task failures. A camera/system notice can be hidden with `:system_messages=off` in Matteo's curses client, and restored with `:system_messages=on`; that choice does not alter execution.

For a failed task, inspect its command ID, planning error, whether motion was attempted, actual held objects and measured final scene. Do not repeatedly resend a movement while its original result is pending. A successful API response from the language provider proves only that a provider request returned, not that the arm task succeeded. The release validation report distinguishes controlled model fixtures from native simulator tests; no physical Panda is commissioned by this update.

32 Session report export and motion reliability (.87)

Build v59.20260919.91; Panda simulation pack revision 20.

32.1 Save, upload and open a report

From the extracted project folder, with the MMSW server running, use an actual UID:

```
python -m vfxrio.cli summary --uid=6 --upload --save --browser
```

UID 6 selects Celestia's visible activity; UID 2 selects Matteo's. Do not type the literal angle-bracket placeholder `<UID>`. Auto-reply and Wake Studio do not need to be enabled to export. The report contains the activity available on the server; it cannot recover deleted history.

The complete transcript and recorded task/call evidence are saved **before AI analysis**, under `vfxrio/server/summary_reports/`. The absolute filename is printed. Analysis updates that file atomically. The total analysis wait is 180 seconds by default; change it with `--analysis-timeout=300`, or skip model calls using `--no-analysis`. An analysis failure or timeout still leaves a usable report. Upload also implies a local save.

Set these in the project `.env` or environment:

```
SFTP_HOST=your-sftp-host
SFTP_USERNAME=your-account
SFTP_PASSWORD=your-password
SFTP_PORT=22
```

The existing destination remains `/cust/0/vfxrio_15201/metame_26564/site/public/The_MMSW`, with public base URL https://metamessage.id34.com/The_MMSW. The SFTP account must have write permission. Missing directories are created when permitted. SFTP connection, authentication and channel operations use a 30-second timeout. On failure, the command reports the error and retains the local file; with `--browser`, it opens that local report. On success, it prints and opens the public report URL. Successful SFTP transfer does not itself test the website's HTTP configuration.

```
python -m vfxrio.cli summary --uid=6 --upload --save --browser \
  --output-dir ./reports \
  --remote-dir /absolute/webroot/The_MMSW \
  --public-url https://your-site.example/The_MMSW \
  --upload-timeout 45
```

Equivalent environment overrides are `MMSW_SUMMARY_DIR`, `MMSW_SUMMARY_REMOTE_DIR`, and `MMSW_SUMMARY_PUBLIC_URL`. They do not change robot-image upload settings.

32.2 Install the index and retain the supplied ASCII artwork

Upload `deploy/summary/index.php` to the **same web directory** as the exported HTML reports, replacing the older index. The web server must support PHP 7.4 or newer. This is a one-time index deployment; the summary command uploads its HTML report and does not silently replace the index.

The index lists reports newest first, with titles, timestamps, UIDs when available, and abstracts. It supports the attached legacy HTML, previous v59 reports, and current metadata. Its lookup uses the index's own directory and needs neither DOM nor mbstring extensions. The report and index

both include the ASCII portrait, logo and attribution from the supplied HTML. The current report template, phone details and full transcript remain available.

32.3 Arm control examples

Request	Executable behavior and evidence
Keep red where it is; stack green on red, then blue on green.	`stack` with `base: current`, ordered bottom to top. The base stays unpicked; every upper support is measured afresh, and base translation/orientation are checked after settling.
Point up, left, right, down, then diagonally up-right; get the red cube.	Raised/sideways empty-hand poses return through a collision-checked joint transition before tabletop handling. No scene or object reset is used.
Repeat those directional poses a little faster.	Optional `point_pose.speed_scale`, 0.25–1.5, default 1. Each pose is rehearsed at that tempo and verified. A saved recipe must be replanned with the new tempo.
Place the held yellow ball in a free table area.	Sphere radius is independent of rotation. Placement uses the measured support height plus radius, followed by release and settling checks.

The default stack-site behavior is retained when `base` is omitted. A fixed base must be resting, and an obstructed or unstable configuration can still fail. Equivalent normalized direction vectors are compared with numerical tolerance, while measured directional error, joint limits, collision checks and payload retention remain required. The new tempo control applies to simulated directional poses, not arbitrary real-hardware speed commands.

32.4 Validation boundary

Regression tests execute native simulated motion and measured verification, plus local report export, controlled SFTP failures and the PHP index with legacy/current HTML fixtures. They do not call a live reasoning provider, connect to the production SFTP account or certify physical-arm behavior. The final test results are recorded in `VALIDATION_v59.87.json`; earlier release evidence remains available separately.

33 MMSW LAB 1: neon lab and industrial workshop (.91)

Two built-in rooms are available around the measured Panda. The original neon laboratory has stainless surfaces, simple panel walls and suspended tubes. The new **Industrial workshop** follows the supplied visual reference: concrete walls, timber benches and ceiling, exposed pipes, layered tool shelves, storage cases, drawers, a wire-mesh partition, workshop equipment, a stool and tool trolley. Cool fluorescent light contrasts with a warm shadowed pendant. Both rooms retain **MMSW LAB 1 – VFXRio & Visgraf** branding and omit computer monitors.

The workshop is a navigable 3D recreation, not an exact photographic scan. The furniture is geometry with perspective and parallax. Original material maps provide timber grain, concrete variation, metal wear, roughness and small surface relief. Its preview is a capture from the actual Robotics Studio renderer. The central worktable preserves the simulator's measured dimensions, which may differ from the reference photograph's foreground table.

33.1 Choose either lab in the ordinary interface

In **Robotics Studio**, find **Lab environment** below **Lighting & boundary**. No Conversation Studio or Expert settings are needed.

1. In **Environment**, choose **Industrial workshop** for the reference-inspired room, timber table, 30% room light and 26% spotlight. Or choose **Neon lab** for the stainless room with 30% room light and 65% spotlight. Both enable rendered view and open **Room view**.
2. Drag to orbit, right-drag to pan and scroll to zoom. The camera stays inside the solid walls. Overview, Top and Front remain available.
3. Adjust **Room light** and **Spotlight** independently. For a dark room with the table lit, click **Room lights off** and leave Spotlight on.
4. Use **Show safe-area boundary** to show or hide the white table strip, and **Hide labels** for joint and object labels.
5. Optionally **Upload new image**: a 2:1 panorama surrounds the room; a normal photo appears on the right wall. The Environment menu lets you choose either mode. **Remove image** restores the built-in lab.

To keep your current materials and lighting, simply check **Show solid 3D lab** and click **Room view**. **Lab lighting** changes only the lights (45% room, 100% spotlight). Uncheck the lab to return to the open environment, or uncheck **Rendered view** for wireframe. The arm pose and task are preserved. Uploads stay on this browser/site address. JPEG, PNG and WebP are supported, up to 20 MB; a photograph supplies surface appearance rather than new furniture.

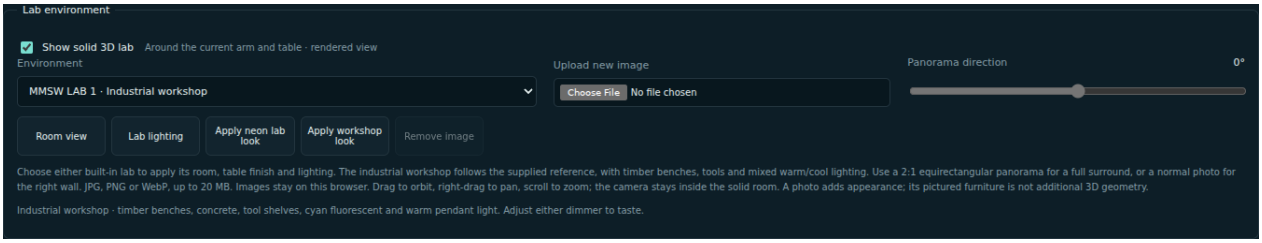


Figure 9: The two built-in rooms and their appearance buttons in .91.

33.2 Industrial workshop appearance

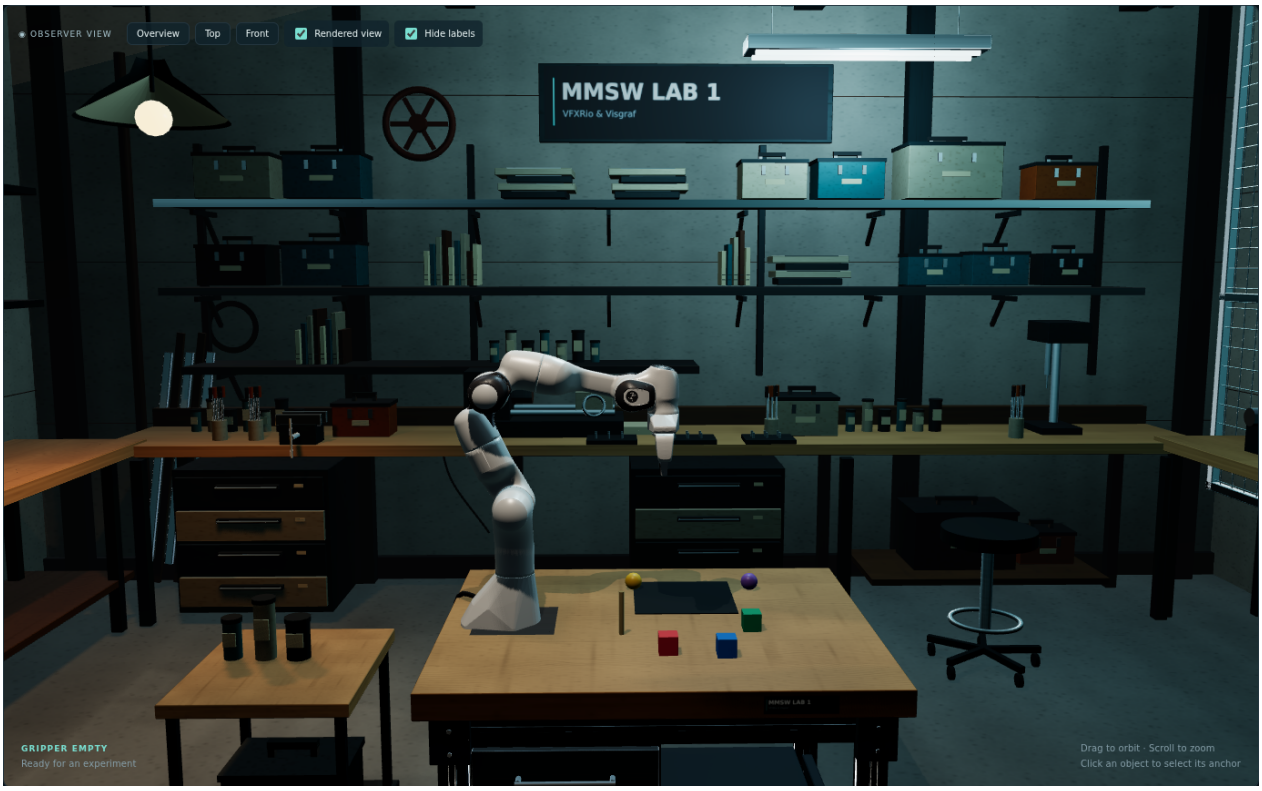


Figure 10: Actual .91 Robotics Studio room view. The industrial workshop uses three-dimensional furniture, textured surfaces and mixed warm/cool lighting.

Control	Result
Environment: Industrial workshop / Apply workshop look	Enables the rendered workshop, Workshop timber, 30% room light, 26% spotlight and its Room view.
Environment: Neon lab / Apply neon lab look	Enables the original rendered lab, stainless finish, 30% room light, 65% spotlight and its Room view.
Show solid 3D lab	Toggles the selected room without resetting its finish or dimmers.
Lab lighting	Changes lights to 45% room and 100% spotlight; keeps the table finish.
Room light	Dims room fill, fluorescent tubes, pendant, sign and reflections.
Spotlight	Independently dims the cone light aimed at the arm and central table.
Table finish	Choose stainless steel, graphite, aluminium, walnut, Workshop timber or classic wood.

Selecting a built-in room applies its appearance preset. Subsequent changes to materials and dimmers are remembered. Reload restores those settings without reapplying the preset. Both sliders at zero darken the physical rendered scene; measured ink, wireframe guides and UI labels remain separate overlays.

33.3 Measured geometry and ongoing tasks

Figure 11: Closer view of the same measured Panda worktable with Workshop timber. The arm base, paper and task objects retain their reported positions.

The tabletop retains its exact reported size, position and top height; its visual bevel stays inside those dimensions. The mounting plate remains flush with its own depth bias from .90, so it does not flicker when the camera moves. Normal depth testing lets the arm and task objects occlude it correctly.

Switching rooms does not home, reset, move or interrupt the robot. Live strokes remain visible on the paper, and measured object motion continues. Room scenery is presentation outside the robot's collision model; it is not additional manipulable equipment. Panda simulation pack revision 20 is retained.

Drag to orbit, right-drag to pan and scroll to zoom. The camera stays inside the opaque walls, floor and ceiling. Overview, Top, Front and automatic observer tours remain available. **Show safe-area boundary** controls only the visible table strip; movement limits remain the same. **Hide labels** controls the joint and object labels independently.

33.4 Original neon lab and uploaded environments

Choose **MMSW LAB 1: Neon lab** to return to the earlier room. Its workstation, drawers, ceiling tubes and stable stainless mounting are retained. Uncheck **Rendered view** for wireframe, or uncheck the lab to return to the open room.

33.5 Upload, persistence and performance

Upload new image accepts JPEG, PNG or WebP up to 20 MB. The browser resizes it within 4096 by 2048 pixels while preserving aspect ratio. A roughly 2:1 image defaults to **Uploaded 360° panorama**; other images default to **Uploaded wall photo**. A panorama covers the closed room shell and hides the built-in furniture. **Panorama direction** rotates it. A normal photo is framed on the right wall of the neon lab, retaining its sign.

Switching to either built-in room keeps the uploaded image stored. Selecting the upload's menu entry returns to it. **Remove image** deletes it and restores the neon room. An unreadable replacement keeps the last valid image. A photograph supplies surface appearance, not scanned furniture depth or an HDR light probe; light already in the photograph remains part of it.

Settings and uploads remain on this browser and site address, including its port. Reload restores the selected workshop independently of a stored upload. A different device/address, private browsing or cleared site data may require uploading again. If storage is unavailable, the status explains that an image is usable only for this visit. No image is sent through MMSW or to an AI provider.

Each room is built once on demand and reused. Stationary furniture is batched by material, and the workshop's reflection map is cached. Repeated room switches reuse geometry and textures. There is no live reflection capture, environment video stream, external texture download or additional robot command. Reduce Observer quality or disable the lab on a slower computer; no frame-rate guarantee was measured on the user's Mac.

Save view PNG includes the current observer room. Robot sensing-camera images and measured task-result sheets remain separate observations and do not inherit this browser's environment.

33.6 Validation and scope

The current browser checks cover dropdown activation, both lab appearances, independent lighting, saved settings, six opaque boundaries, orbit and zoom, wireframe/room toggles, repeated switching without additional GPU resources, upload preservation, live cube updates, visible ink and zero additional MMSW commands. The UI bootstrap regression also covers local and remote startup, camera controls and command routing. Details and actual screenshots are in `VALIDATION_v59.91.json` and `docs/robotics/v59.91/`.

The .90 mounting regression remains documented separately; it is retained evidence, not a newly repeated hardware test. This update changes no arm planning, physical adapter, server API or task verification. Validation uses Chromium with software WebGL; macOS/Firefox and physical hardware were not exercised.

34 Writing, drawing and other retained robotics features

Panda and M3 simulation have native pen skills and their own paper/tool geometry. M2 has a smaller capability pack and must not inherit Panda skills simply because the interface contains an example. The running pack is authoritative.

The tagged forms below address Aster directly. To use Celestia as coordinator, send the corresponding plain-language task privately to her and keep Aster's conversational Auto-reply OFF.

```
#robot_arm Get the pen and write Matteo then park the pen
#robot_arm Get the pen and draw a circle then park the pen
#robot_arm Stack the cubes
#robot_arm Arrange the cubes from left to right: red, blue, green
#robot_arm Put the spheres together and keep them separate from the cubes
#robot_arm Wave goodbye
#robot_arm Clear the paper
```

Use quotes when an inscription could otherwise be parsed as another action. New drawings use available blank space. Clear the sheet only when you want to discard its current ink; additions to existing drawings keep that ink in place (Section 19). Writing success requires new strokes for the requested text, paper bounds and a lifted pen; a picked pen or old ink is not enough.

Pen parking has its own shared, per-pack location. To inspect, teach or reset it:

```
#robot_arm Where is the pen parking space?
#robot_arm Learn pen parking
#robot_arm Learn pen parking at 0.29, -0.22
#robot_arm Reset pen parking
```

Learning the current pen site requires a resting upright pen and an empty gripper. An explicit XY site is validated through native motion and measured placement. These coordinates are for the Panda example; do not reuse them as physical-arm calibration.

35 Preparing for a real Panda

35.1 What is available now

Layer	Status in this delivery
Panda simulation	Native executable pick, placement, rotation, orientation and verification.
Shared action contract	Stable object and pose semantics, explicit units, axes, quaternion order, provenance and cancellation.
Franka pose encoding	`orientation.libfranka_pose` converts an already calibrated base/EE pose to column-major `O_T_EE`. Tested numerically; performs no device I/O.
ROS 2 transport	Existing `Ros2TrajectoryTransport` provides a bounded trajectory/planning transport building block. It is not a complete Panda adapter.
Physical Panda driver	Not included. `panda.PHYSICAL_OPS` stays empty and `DRIVER` stays unset.
Physical Waveshare packs	Existing independent drivers and commissioning gates remain in place. They do not enable physical Panda rotation.

The current `--adapter` hardware choices are Waveshare M2 and M3. `--arm-model panda` selects a simulator, not a physical Panda connection. There is deliberately no launch command here that pretends otherwise.

35.2 Choose the correct Franka software combination

Identify the actual robot generation and installed system/FCI version before choosing libfranka and ROS packages. Official documentation covers both the older FER/Panda and FR3, but that does not mean every current release/package combination supports every generation. Use the official compatibility information for the installed system.[4, 5]

A direct FCI controller belongs on a suitable Linux real-time workstation with a validated robot-network connection. Franka documents a 1 kHz interface and strict timing requirements. The web application, provider calls, image upload and chat transport must remain outside that control loop.[6]

For a ROS-based implementation, select the description, MoveIt configuration, the ros2_control interfaces and gripper actions that actually support the installed Panda. Verify that support before commissioning. The generic ROS 2 example configuration shipped with MMSW is an integration example; it is not a tested Panda deployment profile.

35.3 Calibrate the coordinate frames

The PyBullet tool link and world coordinates are not automatically identical to the physical Franka end-effector and base frames. Let O be the Franka base, W the calibrated task world, TCP the task tool point and EE the frame expected by the hardware interface. Then

$$ {}^{O}T_{EE} = {}^{O}T_{W}\, {}^{W}T_{TCP}\, ({}^{EE}T_{TCP})^{-1}. $$

Measure the base/world and end-effector/tool transforms; validate their units, handedness and rotation convention. Calibrate the camera to the task world, measure the table, object dimensions

and collision geometry, and provide time-stamped physical object poses. The shared conversion helper does not estimate these transforms.

```
from vfxrio.robotics.orientation import libfranka_pose
# Inputs must already be in Franka base / EE coordinates.
O_T_EE = libfranka_pose(base_position_m, base_ee_quaternion_xyzw)
# Encoding only. No robot connection or motion occurs here.
```

35.4 Implement the measured adapter boundary

A physical Panda integration should implement `RobotAdapter`: `validate_plan`, `execute`, `snapshot`, `stop` and `close`. Keep the same high-level action semantics, but replace simulation internals with measured hardware planning and execution.

1. Read actual joint/EE state and calibrated object poses. Advertise only operations that have been implemented and commissioned. Keep `motion_enabled` false during initial connection and observation.
2. Convert requested poses into the physical base/EE frame. Use a collision-aware planner and the actual payload, fingers, table and surrounding geometry.
3. Time-parameterize trajectories using the installed model's joint, Cartesian, acceleration and jerk constraints. Do not send PyBullet waypoints directly as a hardware trajectory.
4. Use the gripper's measured grasp result and physical feedback. The simulator's fixed constraint is not a hardware grasp model. Rotation of a held object requires checking slip and object pose, not only wrist angle.
5. Execute outside the chat/provider process, monitor fresh feedback, honor cancellation during motion, and latch faults when stopping cannot be acknowledged.
6. Verify final object orientation, placement, release and settling from physical measurements. Publish physical evidence sources; never label a hardware result as simulator ground truth.

The Franka interface specifies position, velocity, acceleration and jerk constraints; a valid pose alone is insufficient for a valid trajectory. Use the robot-specific limits and start-state requirements in the official interface specification.[7]

35.5 Commissioning acceptance sequence

Start with read-only state and camera validation. Verify frame transforms with known stationary targets. Test empty-tool translations and small rotations in a clear workspace. Confirm stop/disconnect behavior and controller faults. Next commission the gripper and a supported test object, then lift/turn/place at conservative settings with independent object-pose measurement. Finally validate MMSW routing, records and operator controls while keeping the hardware stop system accessible.

This sequence describes the remaining integration work. It does not certify a physical deployment, and the software changes in this archive do not remove the need for it.

36 Conversational guardrail status

The evidence verifier checks Celestia's and Aster's replies; it does not rewrite the structured arm command. Run `:guardrail status` or `:conf_print` to see the effective verifier model and last check. An empty model override uses the current reply model. Hidden capability packets do not invoke this verifier. An empty or failed verifier response is recorded explicitly, and robot narration falls back to the measured execution record. Successful completion requires both completed status and positive verification. Simulation evidence is not a physical-world completion. See `GUARDRAIL_ROBOTICS_REVIEW_V59.md` for the retained .64 review.

Keep the existing verifier configuration when applying the Celestia-led setup. A blank optional model override is different from an empty verifier response and is not, by itself, a broken robot interface. Neither turning the verifier off nor changing its model is the control for conversational back-and-forth; use Auto-reply, Wake Studio and floor settings in Section 5.

37 Troubleshooting

Symptom	Action and interpretation
Celestia and Aster keep exchanging replies or tasks	Use Stop / cancel motion for active movement. Put Celestia on hold while checking Aster Auto-reply OFF, both Wake Studios OFF, and intended graph routes. Resume with one private request to Celestia.
Aster is on hold: will tasks run?	Yes. Keep its client and adapter connected. Structured commands and measured task results do not require ordinary conversational Auto-reply.
No answer from Celestia	Check Auto-reply ON, the robotics-admin role and both graph directions. With Conversation Studio ON also check Agency ON, floor Auto, no mute, and an actual floor grant.
Studio says Aster Auto-reply is stopped	Expected in A/B; not the intended conversational state in C. Check the selected setup and confirm the client/adapter connections separately.
Auto-reply turns on again	Disable Aster's Wake Studio and check saved profile/runtime settings; a matching wake rule can reactivate a held client.
Two similar commands appear	A multi-step task has one command ID; two IDs can indicate two submissions. Do not send the same task to Celestia and Aster or in both private and public chat.
Left-of-sphere rejected	Use build .67 for every process. Relative destinations are now grounded locally; an older adapter or stale pack must be restarted.
Celestia has old commands	Restore both graph directions and allow the capability renewal. Restart Celestia from the updated project as well as Aster.
Wave fails after a joint move	This release transitions back into the Cartesian task box with measured joint motion. Inspect recovery or collision evidence if it still fails.
"No verified rotation primitive"	Check v59.20260919.91 and Panda revision 20 in the running pack. Stop old server/Studio/client processes, restart from the new directory, and refresh the page.

Symptom	Action and interpretation
“No learn operation”	Use the new native rotation recipe. Learning is controller metadata; the planner should not request a learn motor opcode. Re-teach revision-1 recipes after the pack revision change.
Only the scene cube changes	Use “Pick, turn & park cube”. “Set scene yaw” changes simulation setup without moving the arm.
Wrong turning direction	Check relative versus absolute angle and world versus tool frame. World-Z clockwise viewed from above is negative.
Which cube?	Name blue/red/green or establish an unambiguous held/focused object. Do not expect a random selection.
Unreachable pose / joint limit	Change the task position or request a smaller turn. Full orientation support does not imply every pose is reachable. Inspect held state after a failed approach/turn.
Cube is tilted	Restore an upright orientation before the native tabletop cube operation or placement.
Parking origin unknown	Provide a validated placement destination. Origins cannot be recovered from a guessed camera depth.
Other object held	Put it down before handling a different cube.
No fresh camera / stale feedback	Restore the MMSW connection and sensing stream. A camera frame must return through MMSW; a local preview alone is insufficient.
Missing `websockets` / `uvicorn`	Run core setup in the selected MMSW interpreter. `websocket-client` and `websockets` are distinct distributions.
PyBullet dependency fails	Inspect the robotics interpreter separately. Use the prebuilt macOS Conda backend or a Linux Python version with a compatible wheel.
Pack changed / stale binding	Refresh the received state and submit a new command only after resolving the old command’s recorded outcome.
Hardware connection with Panda	The included hardware selector is Waveshare-specific. Add and commission a measured Panda driver before advertising a Panda physical pack.
Task finished but image URL missing	Inspect the replying client’s existing image-upload configuration. Use the retained task result; do not infer failed motion from upload failure alone.

38 Validation and reproducibility

The retained Guided setup was checked against the .67 launch flags, Auto-reply gate, administrator result correlation, Wake activation, Conversation Studio floor eligibility and open-conversation settings. The .72 report adds spatial drawing, continuous-contact turns, support-aware pushes and bounded evidence projection to coverage of independent fingers, contact-qualified pair grasps, side extraction, payload retention and support verification, as well as measured pushes and short tosses, long fitted writing, persistent PNG storage, local/remote gallery access, vector design, memory migration, result images and transport budgets; a rebuilt manual does not constitute a new live provider or hardware test. Use the session check in Section 5 to confirm your running configuration.

The delivered `VALIDATION_v59.91.json` records browser validation of the detailed laboratory, workbench and lighting. `VALIDATION_v59.87.json` records report export and motion fixes. The retained `VALIDATION_v59.86.json` records the current reliability checks and test limits. The retained `VALIDATION_v59.85.json` records the feedback-mode checks and retained reply regressions: Persona/Mind and language context, evidence fallbacks, result-image suppression, live settings, launch configuration and return routing. The prior `VALIDATION_v59.84.json` records the retained motion and visual-context checks; those motion results are not a new hardware certification. New checks cover continuous-contact surface pulling, upper-load clearing, touching bridge supports, fine cube/sphere pushes, non-contact hand clearance, image-description handoff, source/privacy/reset separation, directional poses, payload parking and retained grasps. Current controller and capability-catalog checks use the full published pack. Relative placement, shared support, side turns, language/Persona replies and embodied farewells remain covered by regression tests. Provider interpretation is exercised with controlled responses; no arbitrary live-model interpretation quality or physical Panda performance is certified.

Run the targeted regression suite in a Python environment containing the project and robotics dependencies:

```
python -m pytest -q tests/test_robotics_obstacles_v79.py \
  tests/test_robotics_spatial_v72.py \
  tests/test_robotics_grasp_v71.py \
  tests/test_robotics_contact_gallery_v70.py \
  tests/test_robotics_art_v69.py \
  tests/test_robotics_dialogue_v67.py \
  tests/test_robotics_joints_v65.py \
  tests/test_robotics_rotation_v63.py
python scripts/validate_robotics_dialogue_v67.py \
  --output /tmp/mmsw-dialogue-validation-67
```

The second command requires Node with built-in WebSocket support (Node 22 or newer is suitable) and core MMSW dependencies. Choose a new output directory. It starts and stops isolated test processes; it does not reuse a production server. The report covers all seven articulation buttons, absolute joint poses, natural joint sequences, carried-cube parking, production transport, second-viewer feedback, and retained cube/tool rotation. The DOM fixtures do not constitute a browser/GPU rendering certification. The live checks use isolated local services, not physical machines or real hardware.

38.1 Regression scenarios

The rotation tests exercise the reported -70-degree turn, repeated positive small turns after that rotation, park-back geometry, three world axes and three local tool axes, quaternion pose commands, invalid/ambiguous input, unsupported pack gating, cancellation, false success evidence, SDK/provider schema and verified phrase memory. Existing robotics tests cover writing, parking, grasp alignment, gravity, cameras, administration and backend behavior.

39 Source map and Overleaf use

File or area	Responsibility
`client/robotics_delegate.py`	Celestia task dispatch, command correlation and result return to the human.
`client/robotics_behavior.py`	Aster task-result narration, including when ordinary Auto-reply is OFF.
`client/messaging.py`, `client/state.py`	Conversational Auto-reply, result suppression and Studio floor eligibility.
`robotics_capabilities.py`	Hidden manifest, fingerprint, replacement and expiry.
`client/robotics_delegate.py`	Administrator translation, conversation context and result correlation.
`robotics_language.py`	Bounded follow-ups and reusable-phrase rules.
`robotics/gripper_control.py`	Independent fingers, combined contact grasp, all-payload release and retention evidence.
`robotics/side_grasp.py`	Side approach, horizontal extraction and measured neighbour outcomes.
`robotics/packs/panda.py`	Panda revision, enabled operations, examples and native rules.
`robotics/planning.py`	Action fields, bounds, provider instruction and result verification.
`robotics/rotation_intent.py`	Complete-clause natural rotation interpretation.
`robotics/semantic_goals.py`, `sdk.py`	Symbolic goal resolution and executable provider schema.
`robotics/orientation.py`	Quaternion/frame math, evidence checks and Franka pose encoding.
`robotics/simulator.py`	Native Panda motion and measured rotation/placement.
`robotics/artwork.py`, `robotics/sketch.py`	General design brief, declarative curves and local trace compilation.
`robotics/pen_styles.py`	Original cursive strokes and bounded lettering parameters.
`robotics/result_views.py`	Bounded dark task sheet from measured simulation geometry.
`robotics/contact_motion.py`	Motor/contact pushes and measured moving-release tosses.
`robotics/drawing_archive.py`	Persistent measured PNG sheets, thumbnails and metadata.
`robotics/controller.py`	Queue, routing, ownership, camera checkpoints, records and learning.
`robotics/static/`	Studio controls and received-state presentation.
`tests/test_robotics_rotation_v63.py`	Contract and native-physics regressions.
`scripts/validate_robotics_rotation_v63.py`	Isolated live MMSW transport validation.

Paths beginning with `client/` or `robotics/` in the table are relative to `vfxrio/`. Older versioned notes remain in the distribution for history; use this manual and `ROBOTICS_ROTATION_V59.md` for the retained cube/tool rotation behavior, and `ROBOTICS_JOINTS_V59.md` for articulation control. For the Celestia-led operating configuration, use documentation revision 30 of this manual; an older

manual bundled inside an earlier download may still contain the broader Auto-reply recommendation.

The new spatial geometry and contact-turn executors are `robotics/drawing_space.py` and `robotics/surface_rotation.py`; their integration tests are `tests/test_robotics_spatial_v72.py`.

Upload the supplied Overleaf ZIP using **New Project / Upload Project**. Select `main.tex` as the main document and **pdfLaTeX** as compiler. Use a recent TeX Live version. Compile twice for the table of contents and references. All bibliography entries are included in the source; no BibTeX service, remote image or shell escape is required. Upload `setup-addendum.tex` and the complete `images/` folder with `main.tex`; the illustrated appendix depends on these local files. The alternative main file `setup-addendum-standalone.tex` compiles only the addendum.

```
pdflatex -interaction=nonstopmode -halt-on-error main.tex
pdflatex -interaction=nonstopmode -halt-on-error main.tex
```

40 Digital Twin scope and migration to .91

40.1 What the Digital Twin represents

This project uses the term **Digital Twin of the Franka Panda** for the experimental virtual counterpart of the seven-joint arm, gripper and tabletop workspace. The executable state comes from the Panda simulation; Robotics Studio presents that state as wireframe geometry or a solid rendered scene. Human instructions, Celestia's mediation and Aster's task execution form a closed software loop with measured simulation feedback.

The .91 source declares `REVISION=20`, `DRIVER=None` and `PHYSICAL_OPS=()` in `vfxrio/robotics/packs/panda.py`. Consequently, this delivered build does not implement a measured physical Panda connection. "Digital Twin" in this manual identifies the experiment and its integration architecture; it does not assert a commissioned, continuously synchronised physical/virtual installation. The hardware preparation chapter remains the reference for a future physical adapter.

Layer	Current responsibility	Evidence boundary
Intent and dialogue	Human request, Celestia delegation and Aster task context	A request or verbal answer does not establish successful motion.
Execution	Panda native simulation, inverse kinematics and contact/pose checks	Measurements are simulator observations, not physical sensor readings.
Observer	Received arm/object poses, live ink, views and room appearance	A camera angle or room change is presentation, not an execution command.
Task record	Before/after observations, verification and saved images	Match the record to the task and build before making numerical claims.
Physical integration	Calibrated frame conversion and documented adapter boundary	Device I/O, commissioning and physical performance remain outside this release.

40.2 Update from the supplied .89 manual

1. Follow the installation chapter using the complete `MMSW-v59-complete-v59.20260919.91.zip`. Restart every process from the same extracted root; refreshing a browser alone cannot update an adapter.
2. Confirm build v59.20260919.91 and `panda.simulation` revision 20. Do not use the older revision-14 text from the .89 manual as the expected value.
3. Keep the intended dialogue routing. For Celestia-led operation, Celestia Auto-reply is ON and Aster Auto-reply is OFF; the delegated robotics result path still delivers task results. Use the full setup appendix for graph edges and roles.
4. Open the Lab environment controls. Choose **Industrial workshop** or **Apply workshop look**. The workshop preset selects timber, 30% room light and 26% spotlight. **Neon lab** selects stainless steel, 30% room light and 65% spotlight. Later slider changes persist independently.
5. In .90 and .91, the old decorative monitor is removed and the flush mounting surface is stabilised. An older .89 screenshot showing a computer is historical, not a missing control in the new interface.

6. Use the task record and measured paper view when evaluating a drawing. Save view PNG captures the observer's presentation; it is not interchangeable with a sensing-camera frame or a measured task-result sheet.

40.3 Reproducible writing and drawing review

For a demonstration, record the build, pack revision, requester, task identifier, instruction, initial objects and paper content. Retain the final task JSON, measured paper PNG and multi-view sheet. Record any planner/provider configuration needed to repeat interpretation, without publishing secrets.

The supplied examples cover cursive writing, longer fitted lettering, a Brillo box illustration, additive drawing and an abstract aurora. An instruction such as "get the pen, draw, then park it" includes both content creation and a final state. Review both: visible ink does not by itself verify parking. Likewise, the additive-drawing regression requires comparison of the original stroke data; a before/after JPEG is an illustration of that check, not a replacement for it.

For the aurora demonstration, the supplied instruction explicitly requests `obstacle_policy clear`. Consult the obstacle-policy chapter for the simulation's measured workspace preparation. This option is not permission to ignore hardware collisions. No drawing demonstration here establishes physical contact pressure, friction calibration or real-world path accuracy.

40.4 Validation provenance

The archived .91 validation reports seven browser check groups and four Studio bootstrap variants passing. The renderer was Chromium 153 with SwiftShader and a native simulation-state fixture. Those records cover appearance switching, lighting, persistence, uploads, boundaries, live object/ink updates and the absence of extra MMSW commands from environment controls. They are retained release evidence, not motion or browser tests rerun for this documentation revision.

Documentation revision 30 compiles and reviews the manual and preserves the 16 supplied JPEGs byte-for-byte. It does not create a new software build. The separate VFXRio publication uses an editorial hero with generative interface removal. That hero is deliberately excluded from this manual's experiment atlas.

Record	Interpretation
`VALIDATION_v59.91.json`	Current release's browser checks and limits.
.90 mounting regression	Retained older evidence; not counted as a new .91 test.
Supplied JPEG figures	Visual records with original UI/captions; not numerical logs.
Source hash manifest	Connects each reproduced figure to its exact uploaded bytes.
Future physical experiment	Requires its own calibrated driver, measurements and acceptance record.

41 Experiment image atlas

The following pages reproduce the 16 images supplied for this edition. Dates come from the supplied filenames and are not authenticated execution timestamps. Existing captions and UI overlays remain intact. These examples illustrate the experiment's development; they are not all labelled as fresh .91 tests.

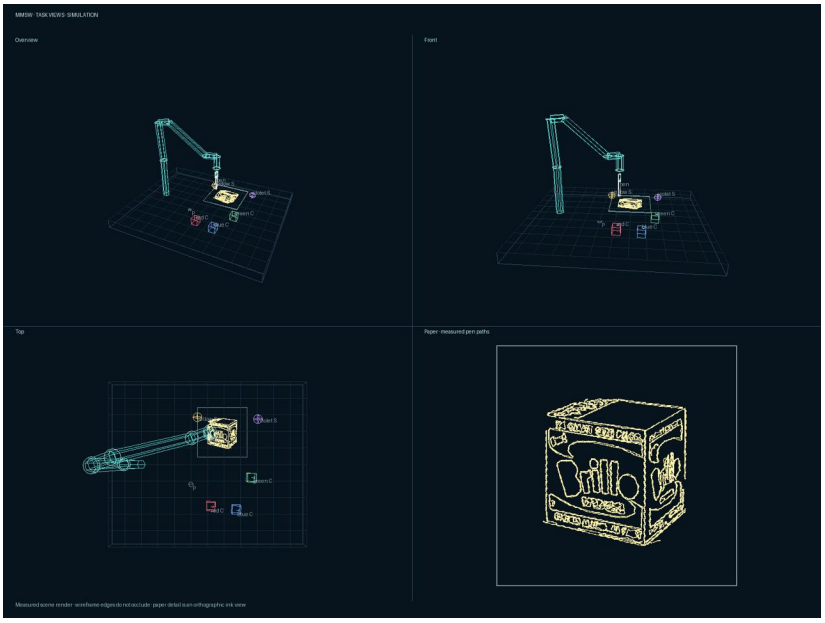


Plate 1. Four-view sheet of a Brillo box drawing. The paper panel displays recorded simulation pen paths; the image alone does not establish physical execution.

Source: 2026-09-17 / 20.27.49 (supplied filename).

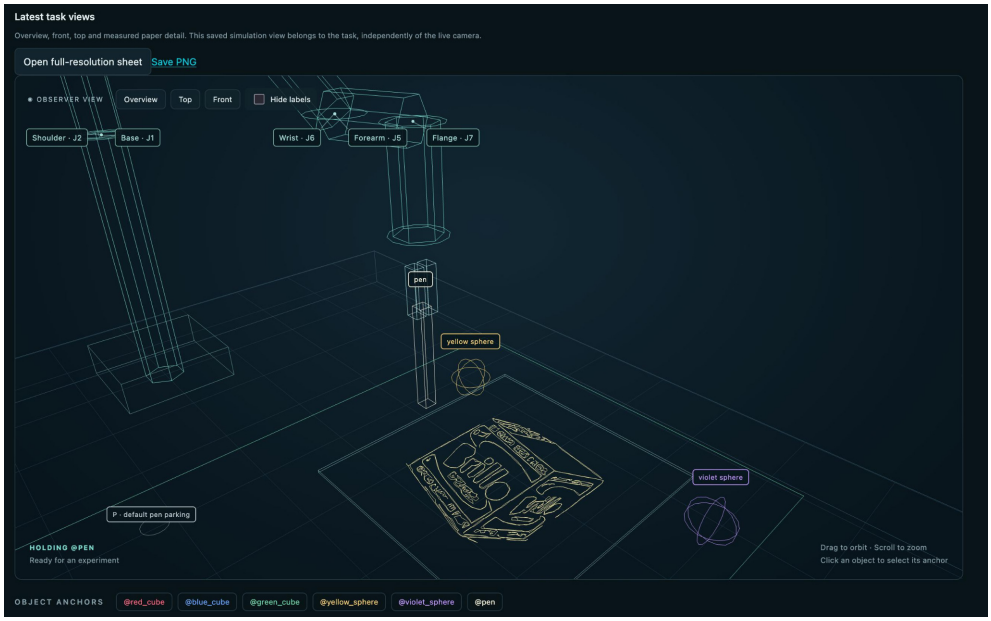


Plate 2. Wireframe Observer with the pen, named object anchors and Brillo box drawing. This is an observer view, not a new camera measurement.

Source: 2026-09-17 / 20.35.11 (supplied filename).

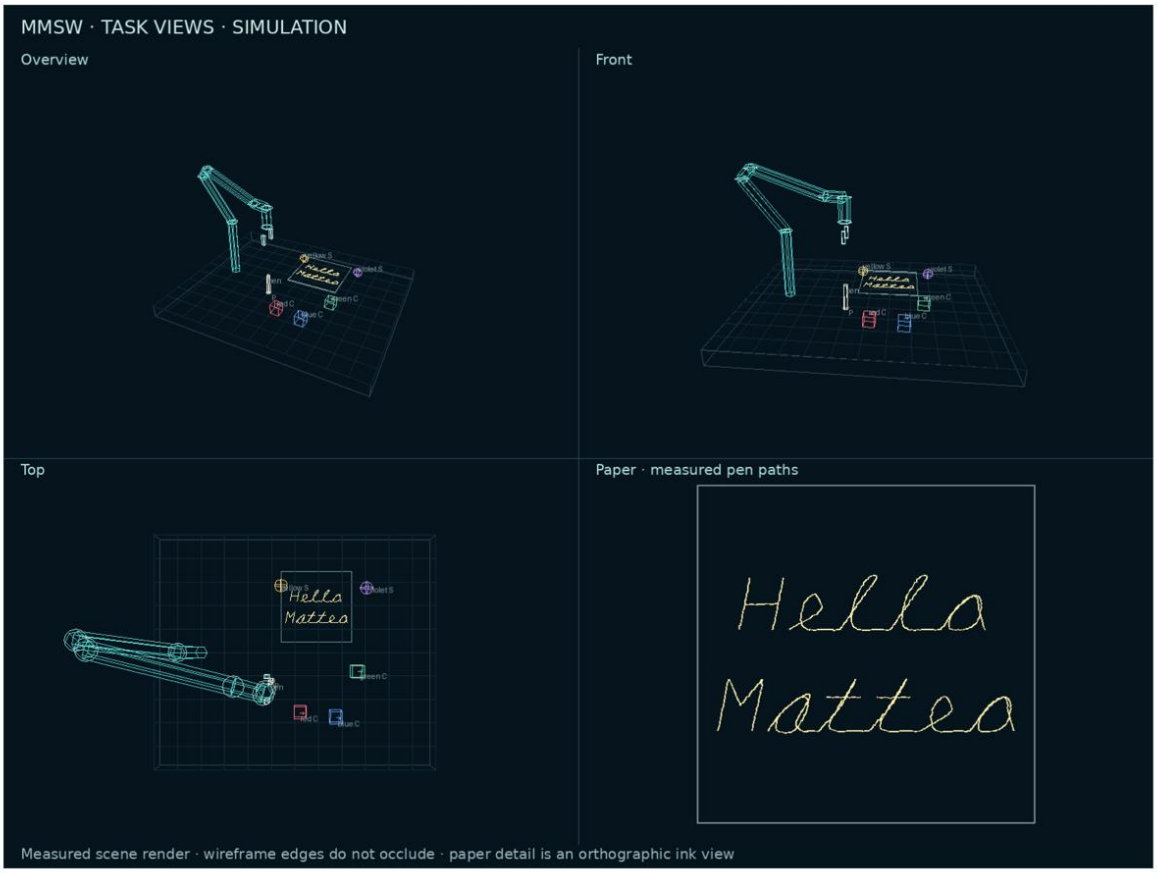


Figure 1: An actual simulated cursive writing result. The arm traced “Hello Matteo” and parked the pen. The sheet uses measured scene geometry and recorded ink. This example’s PNG is 33,896 bytes; future tasks may differ.

Plate 3. Supplied cursive-writing result, including its original caption. The caption identifies a simulated execution and recorded ink.

Source: 2026-09-17 / 20.38.37 (supplied filename).

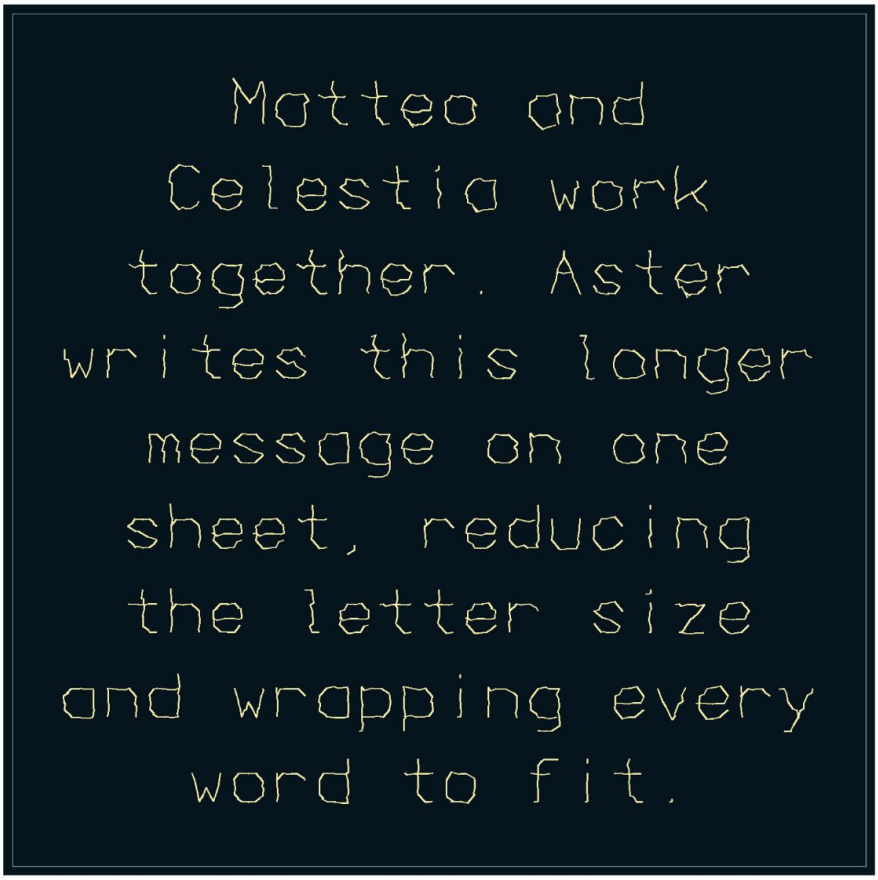
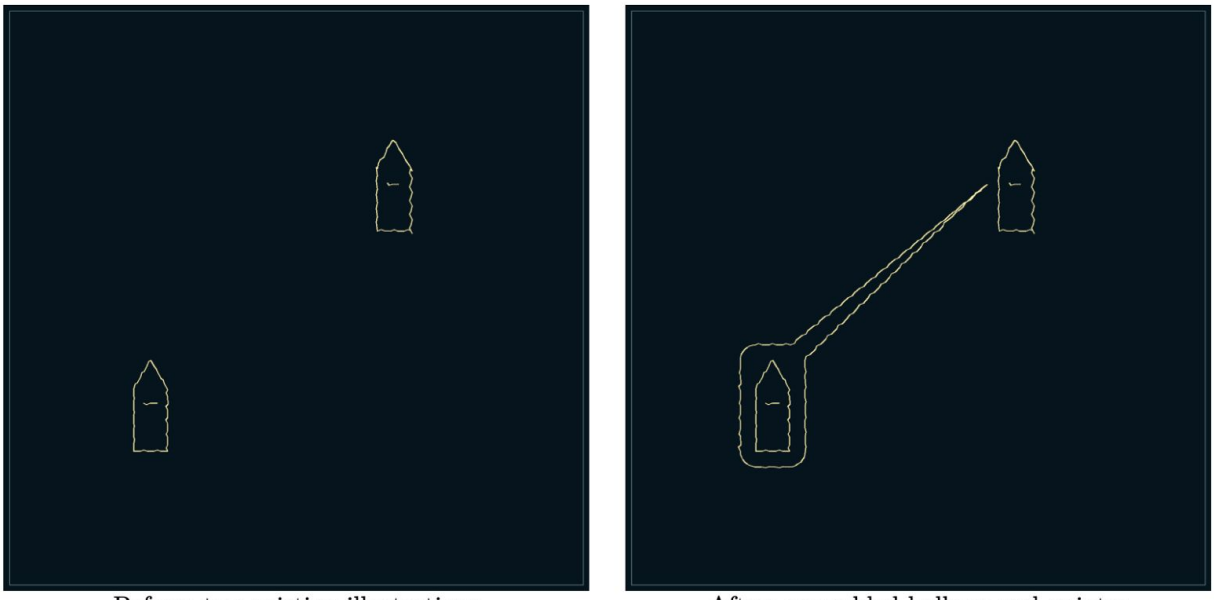


Figure 2: Actual recorded Panda pen paths, automatically fitted and wrapped. The 1600 × 1600 archived PNG is rendered on the same dark background as the task views. It is not a preview of unexecuted letter geometry.

Plate 4. Supplied fitted-writing result, retaining its original caption. The stated 1600 by 1600 archive resolution describes the original PNG, not this JPEG reproduction.

Source: 2026-09-17 / 20.39.03 (supplied filename).



Before: two existing illustrations

After: one added balloon and pointer

Figure 3: Native simulation regression using two simple test illustrations. The original measured strokes are byte-for-byte unchanged. Only the enclosing outline and its tail were added; neither subject was copied. This is a geometry test, not a claim of artistic likeness.

Plate 5. Supplied before/after regression figure. Its original caption reports byte-for-byte preservation of earlier strokes. This publication does not independently recompute that comparison from the JPEG.

Source: 2026-09-17 / 20.39.53 (supplied filename).

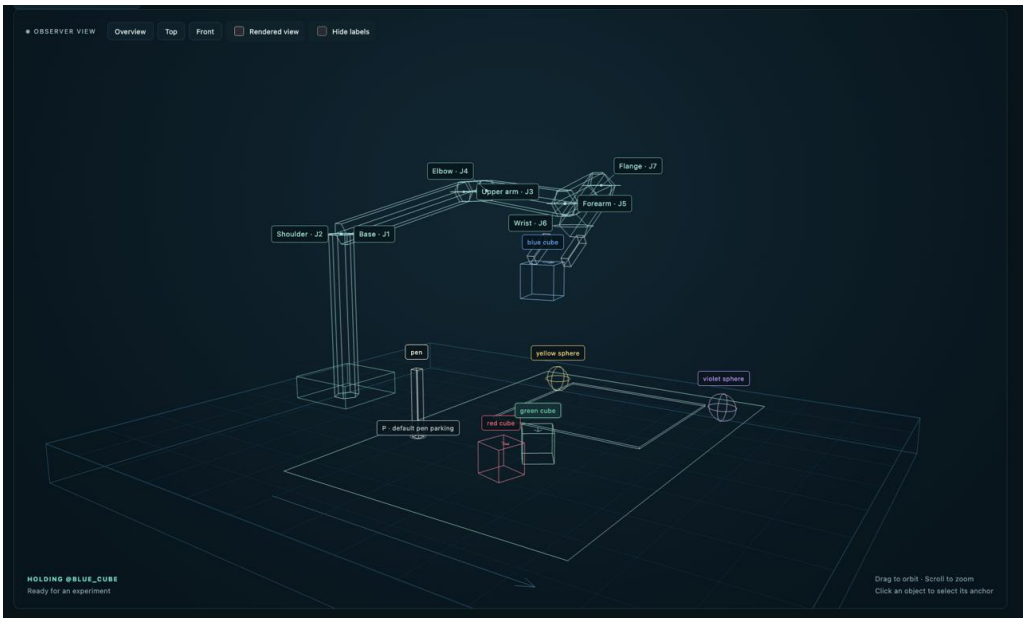


Plate 6. Wireframe view with the blue cube held and joint/object labels visible. A single still is not a trajectory or contact validation.

Source: 2026-09-18 / 08.45.59 (supplied filename).

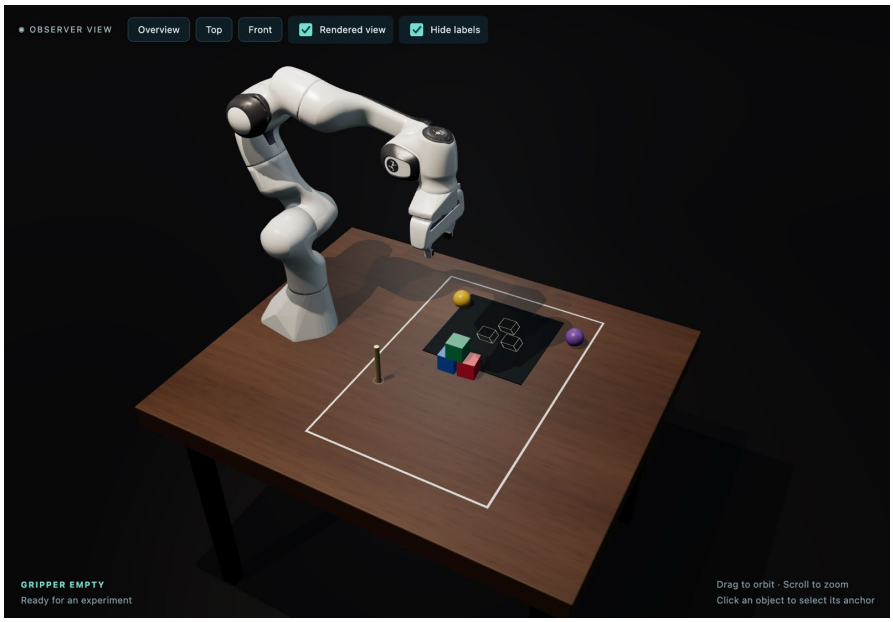


Plate 7. Rendered workspace with coloured cubes, reference spheres, pen and drawing sheet. Lighting and surface appearance belong to the presentation layer.

Source: 2026-09-18 / 20.53.04 (supplied filename).

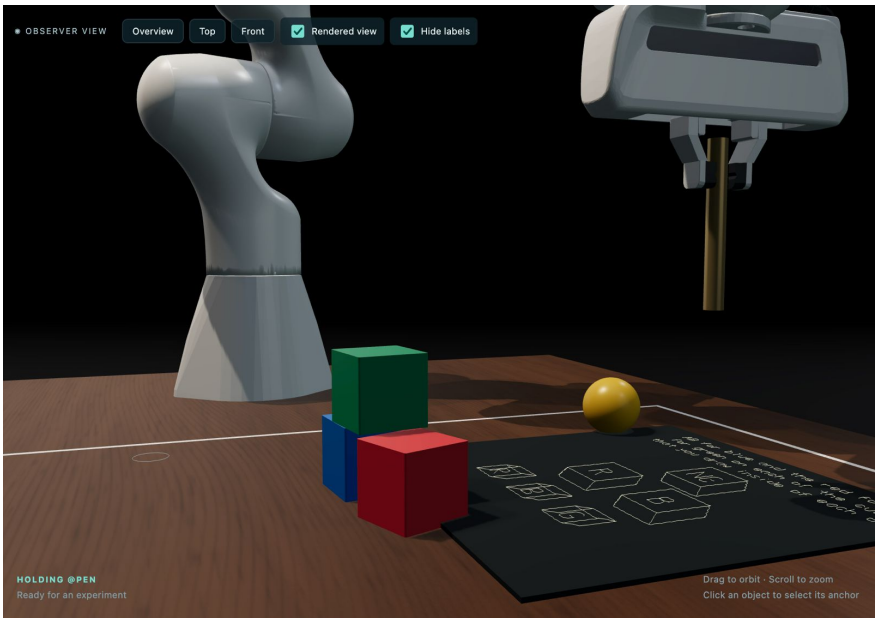


Plate 8. Close view of the gripper holding the pen above the simulated tabletop. This image does not measure physical pen pressure or gripping force.

Source: 2026-09-18 / 21.04.13 (supplied filename).

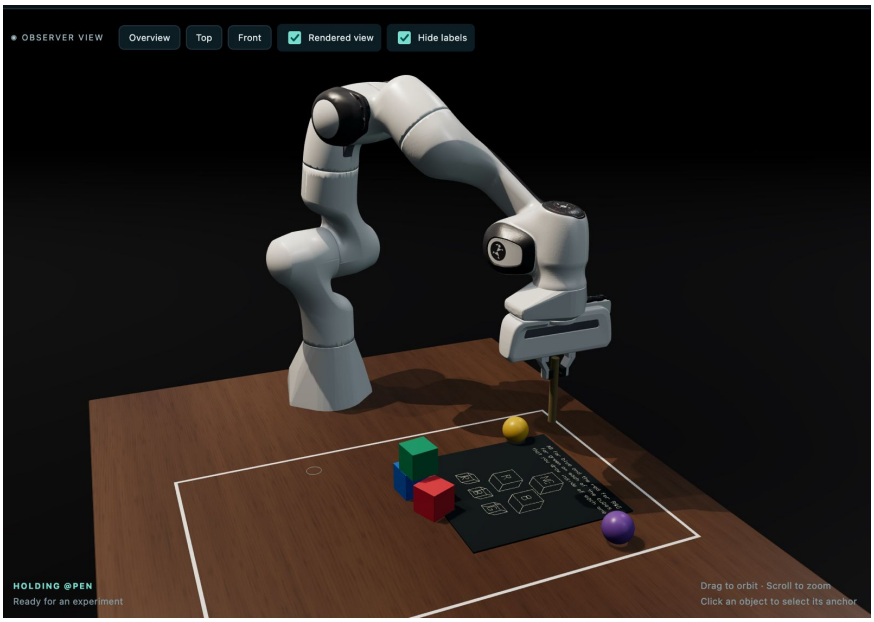


Plate 9. Rendered overview of a pen task with coloured cubes and writing. Use the corresponding task record to establish its final status.

Source: 2026-09-18 / 21.04.39 (supplied filename).

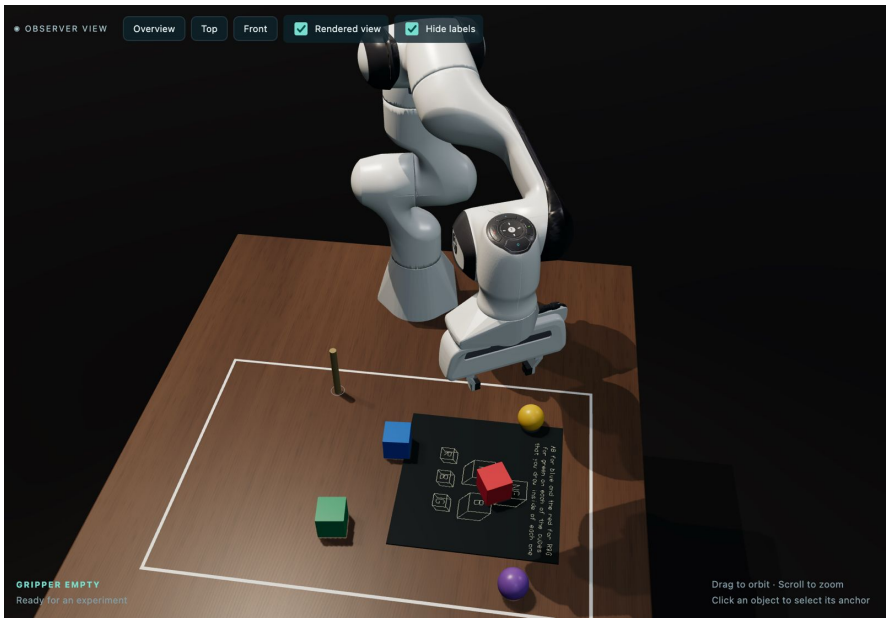


Plate 10. Oblique top view of objects and ink. Screen-space appearance is not a calibrated measurement of clearance.

Source: 2026-09-18 / 21.13.14 (supplied filename).

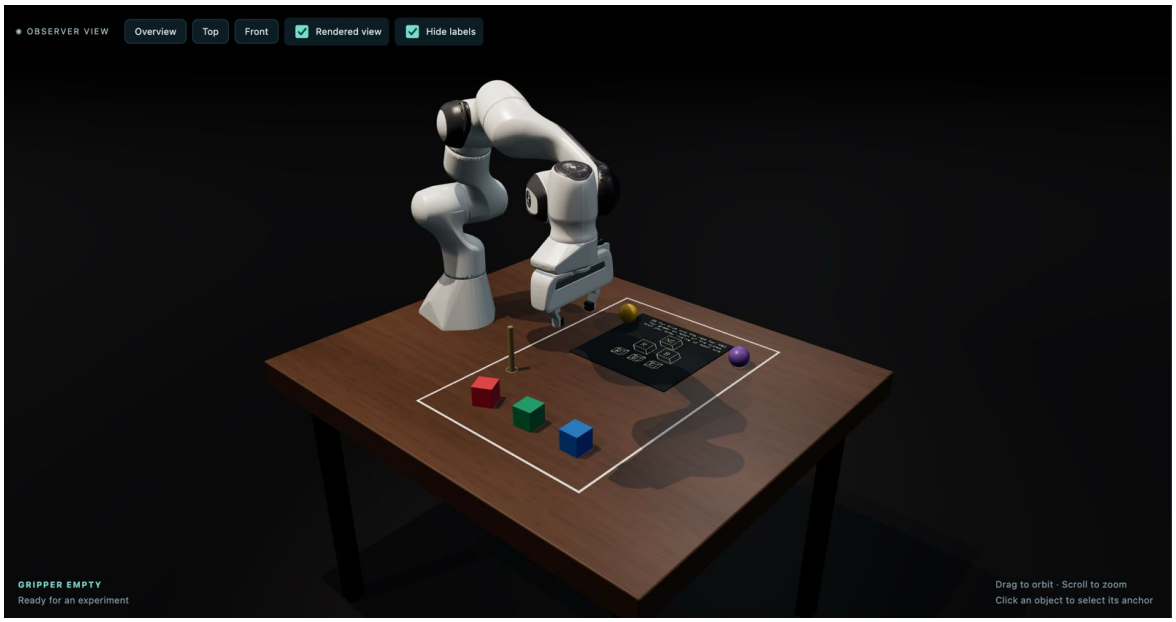


Plate 11. Rendered arrangement showing the parked pen and separated cubes. The screenshot reports an empty gripper; inspect task evidence for release and settling.

Source: 2026-09-18 / 23.08.54 (supplied filename).

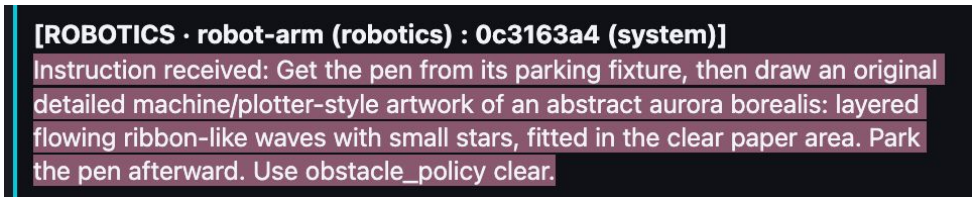


Plate 12. User-supplied instruction record for the aurora task. A received instruction records intent, not successful completion.

Source: 2026-09-19 / 14.45.28 (supplied filename).

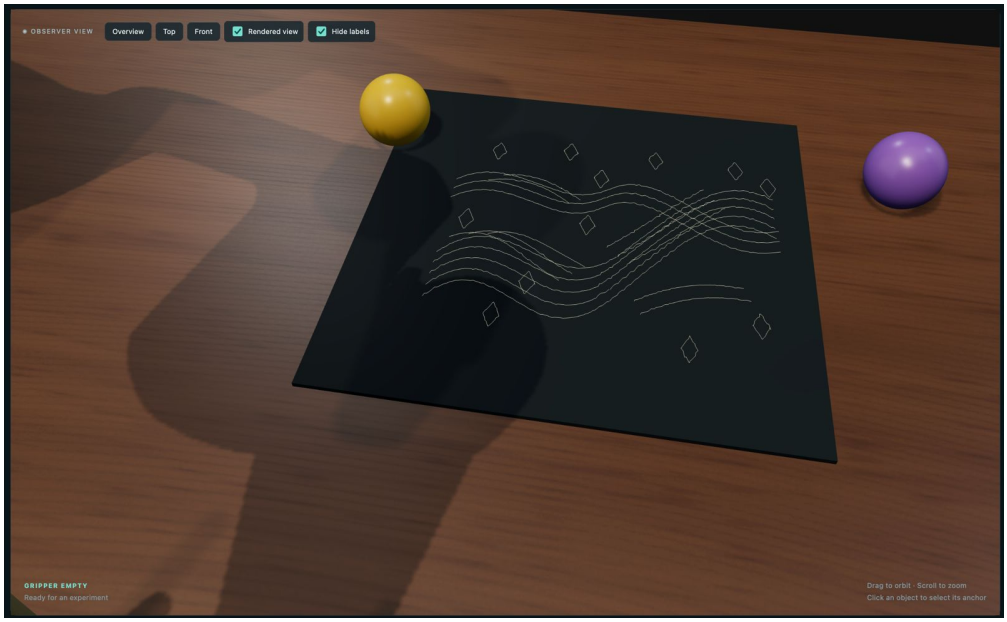


Plate 13. Rendered view of the abstract aurora drawing. This view presents the simulated ink under scene lighting.

Source: 2026-09-19 / 14.45.02 (supplied filename).

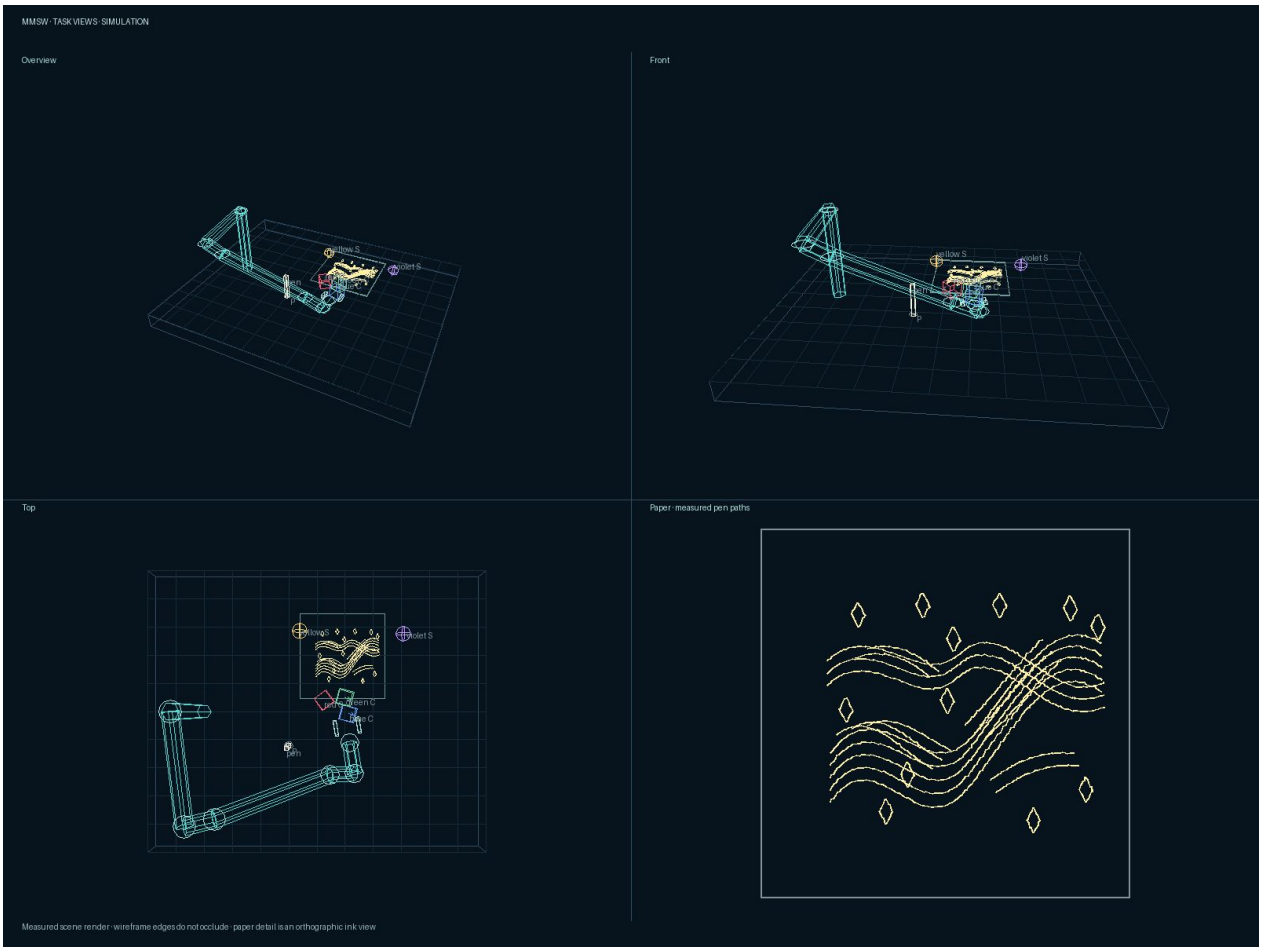


Plate 14. Four-view aurora task sheet, including the recorded paper paths. Retain the original task JSON and archive PNG for numerical reproduction.

Source: 2026-09-19 / 14.56.22 (supplied filename).

Plate 15. User-supplied laboratory view with VFXRio and Visgraf branding. The room is a visual environment; its furniture is not added to the simulation collision world.

Source: 2026-09-20 / 00.18.40 (supplied filename).

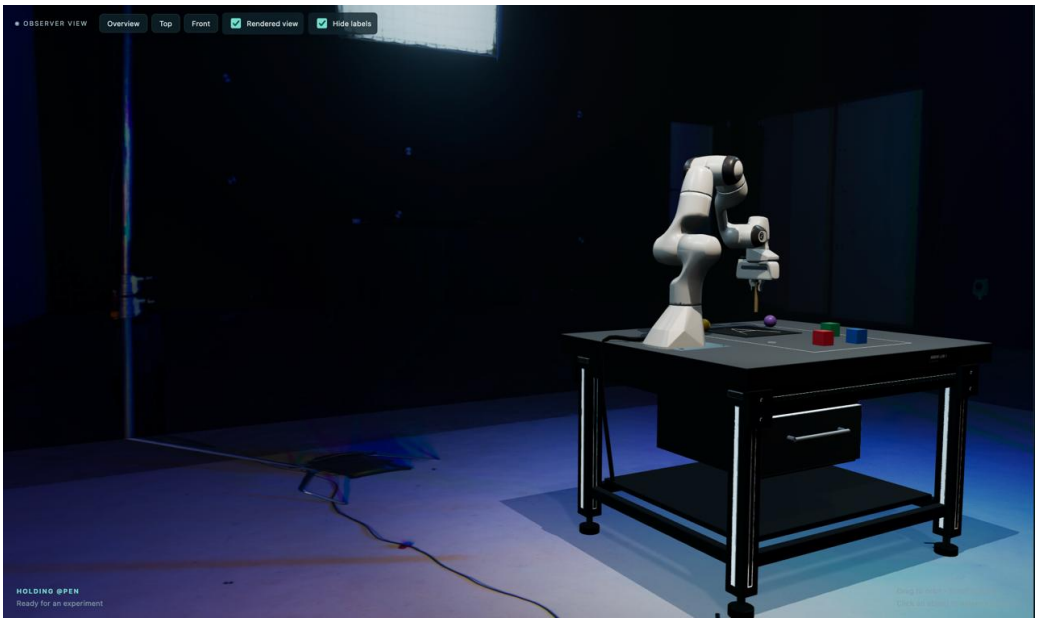


Plate 16. User-supplied wide laboratory view, with the original Observer controls preserved. The image does not identify the selected environment mode or prove physical telemetry.

Source: 2026-09-20 / 03.18.03 (supplied filename).

A Illustrated setup addendum: Celestia coordinates Aster

MMSW v59.20260919.91 / Conversation Studio v8.2.2 / documentation revision 30. This addendum gives a simple setup using **Guided mode**, plus each client's Auto-reply and Wake controls. Expert is optional reference only. The goal is one human request to Celestia, one validated delegated task through Aster, and a measured result returned to the requester.

A.1 New in .85: optional characterful feedback

In **Celestia's HTML5 client**, open the sidebar, expand **Robots · sensing camera**, then check **Expressive robotics feedback** below the camera/components information. No Expert mode is needed. Load the desired Persona and skills in Celestia's Persona Studio; enable State of Mind in her Mind Studio if wanted. Disabled features stay disabled.

Switch	Feedback style	Result-image link
OFF (default)	Existing feedback pipeline	Included when available
ON	Enabled Persona/Mind shapes character and perspective; measured facts remain authoritative	Omitted from Celestia's reply; task evidence stays in Robotics Studio

The checkbox lasts until restart. Equivalent local commands in Celestia's input are `:robotics_feedback=on`, `:robotics_feedback=off`, and `:robotics_feedback` for status. Launch with `--robotics-feedback on` to start enabled, or add `"robotics_feedback": "on"` to her client JSON configuration. This controls Celestia's delegated feedback, not Aster's independent replies or motion. Keep Aster Auto-reply OFF for the setup below. The full manual's Section 30 explains timing, examples and fallback behavior.

A.2 Retained from .84: describe, then copy the direction

Send Celestia an image and ask what she sees. After she describes an upward-pointing arm, say "Mimic that direction with your robotic arm." The follow-up retains the description from the same conversation. A direct "Copy the arm direction in this image: [URL]" also uses the existing image reader. The arm makes a measured directional approximation and holds the pose; it cannot reproduce an individual human finger. Default behavior gently parks any held payload first. Describing a picture alone never moves the arm.

The same settings also support small surface pulls, empty-hand approach, sphere pushes and a cube placed across two neighbouring cube tops. For example: "Pull red 3 cm toward the robot without lifting, then keep holding it." Default clearing parks an upper load first. Explicit simulation permission to let a stack fall applies only to that task. Ask to "lift the entire stack by the bottom cube" when you want the upper cubes carried too.

A.3 Language, active character and goodbye

A genuine farewell can safely park the payload, verify an empty hand and wave. Keep Celestia Auto-reply ON, Aster OFF and Wake OFF. Reply language follows the human conversation; active Persona and Mind remain in the reply path. Load the desired profile in Celestia's Persona Studio and activate Mind if wanted; disabled features stay disabled. No Expert control is needed for this setup.

A.4 The simple route: Guided, with Aster as the executor

Restart all processes from the same .91 directory and confirm v59.20260919.91 and `panda.simulation revision 20`. The hidden catalog refreshes automatically. The 96 KB capability-message limit is respected without dropping commands.

Setting	Celestia	Aster	Matteo
Auto-reply	**ON**	**OFF**	OFF
Wake Studio	OFF	OFF	Not required
Guided Session role	Lead / facilitator	Excluded	Human guide
Guided Open conversation	0: Direct / assigned only	Not needed	Not applicable

Room: Studio ON; Collaborative work; Room energy: Keep current Entity settings; Fixed primary host; Primary host: Celestia Borealis; Temporary facilitator: None.

Quick setup in six steps

1. Keep the server, Robotics Studio, Aster, Celestia and Matteo running. Start Celestia with `--robotics-admin robot-arm`.
2. Connect Matteo ↔ Celestia and Celestia ↔ Aster. Both arrows must exist for each pair.
3. In Guided, choose the room values above. Set Celestia's role to **Lead / facilitator**, her Open conversation to **0: Direct / assigned only**, and Aster's role to **Excluded**.
4. Review the preview, confirm Celestia is the host, then click **Configure room**.
5. In each AI client's own page, save Wake Studio OFF. Then start Celestia's Auto-reply and stop Aster's Auto-reply.
6. Refresh room status. Give Celestia one bounded task and wait for its measured outcome before submitting another.

Excluded means excluded from Studio speaking turns. It does not disconnect Aster, remove graph edges or switch off its robotics adapter. In .67 it sets the Studio conversation mute. Validated robotics commands and results use a separate route. Keep Aster and Robotics Studio running.

How this relates to Setup B's earlier table. The main manual also gives an explicit metadata variant: Agency OFF, floor off and Can host OFF for Aster. Guided does not expose those individual switches. This simpler Guided variant uses speaking-role exclusion plus Auto-reply OFF. It does *not* claim to change Aster's hidden floor/Agency fields: a remaining floor auto label is not a failure when Excluded/MUTED and Auto-reply OFF are confirmed. Celestia's Lead role supplies Agency ON, floor auto and host eligibility.

The supplied screenshots show a session being configured. Gold outlines locate controls; the text describes what to select. They are not evidence that the pictured settings already match this guide. All original images used in the figures are included in the project.

A.5 Prepare the processes and conversation routes

Finish or cancel an active task, then stop the old server, adapter/Studio and clients in their own terminals before switching installations. Extract the complete updated package and use that same project root in every terminal. If dependencies need repair, run:

```
sh scripts/setup-mmsw-v59.sh --role all
sh scripts/setup-robotics-v59.sh
```

Run each numbered block in a **separate terminal**, leaving the process running:

```
# Terminal 1: server
sh scripts/run-server-v59.sh

# Terminal 2: Robotics Studio, adapter and Panda simulator
sh scripts/run-robotics-studio-v59.sh \
  --server http://127.0.0.1:9100 --arm-model panda

# Terminal 3: Aster executes tasks; ordinary conversation is OFF
sh scripts/run-robot-v59.sh 127.0.0.1 \
  --mode none --agency off --conversation-floor off --can-host off

# Terminal 4: Celestia interprets and delegates to Aster
sh scripts/run-celestia-v59.sh 127.0.0.1 \
  --robotics-admin robot-arm --mode auto \
  --agency on --conversation-floor auto --can-host on

# Terminal 5: Matteo is the human operator
sh scripts/run-matteo-v59.sh 127.0.0.1 \
  --mode none --agency off --conversation-floor manual --can-host off
```

These commands select the intended client defaults. A saved Studio override or Wake profile can still affect the live session; verify the interface after startup.

Route	Purpose
Matteo ↔ Celestia	Deliver the human request and Celestia's response in both directions.
Celestia ↔ Aster	Delegate tasks, receive results and exchange hidden capabilities.
Matteo ↔ Aster, optional	Allow direct public delivery between them. This is an additional direct command route.

Check both directed edges of each required pair in the MMSW connection graph. Conversation Studio does not add them. A chain supports mediated tasks; it does not make every participant hear every public message automatically.

Open Conversation Studio at `http://127.0.0.1:9100/conversation-studio` and Robotics Studio at `http://127.0.0.1:9100/robotics/?robot=robot-arm`. Open Celestia's and Aster's own client pages using the URLs printed by their launchers. Their personal controls are not on the shared server's Conversation Studio page. For a remote server, replace loopback with its actual host.

A.6 Guided steps 1 and 2: choose the room and fix the host

Start in **Guided**. The next two pages contain the complete room procedure. There is no need to open Expert for this method.

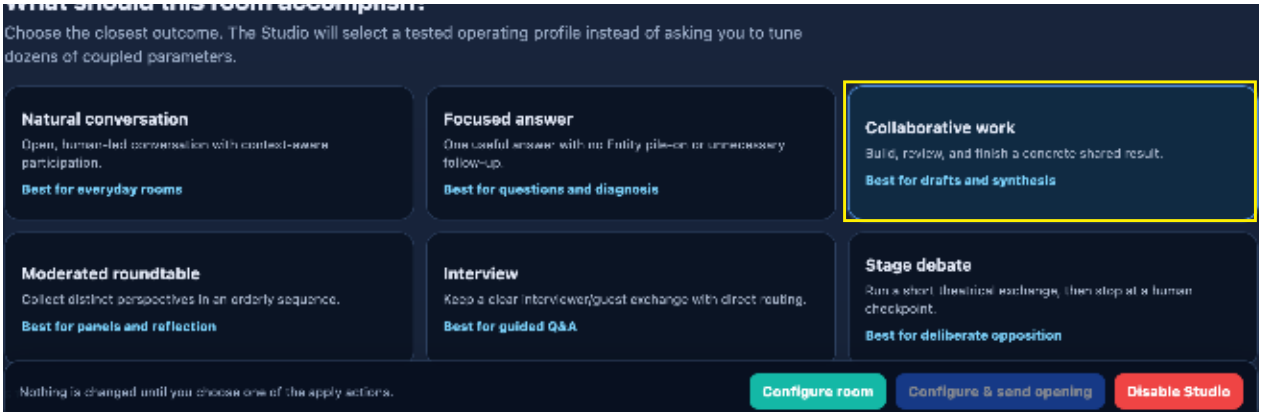


Figure A.1. Step 1: choose **Collaborative work**. The gold outline locates the outcome card.

1. Under **What should this room accomplish?**, choose **Collaborative work**.
2. Under **Give the room a brief**, click the **Collaborative session** preset if desired. Then set **Room energy** to **Keep current Entity settings**. This avoids applying a broad energy profile to the robot executor.
3. In **Who coordinates the room?**, select **Fixed primary**, verify **Plan host policy: Fixed primary host**, choose **Primary host: Celestia Borealis**, and leave **Temporary facilitator: None**.

Figure A.2. Host controls. In the supplied screenshot the Primary host dropdown still says **Choose automatically**; replace that selection with **Celestia Borealis**.

Order matters. In this UI, clicking **Collaborative session** initially selects a temporary-facilitator host mode and lively room energy. Therefore choose the preset *before* restoring Fixed primary host and Keep current Entity settings. Recheck those fields whenever you reapply a preset.

Live state versus draft. The top status cards may already name Celestia while the draft dropdown says Choose automatically. The preview is what the next **Configure room** will apply. Explicitly selecting Celestia prevents automatic host selection from choosing Matteo instead.

A.7 Guided steps 3 and 4: assign the roles and apply

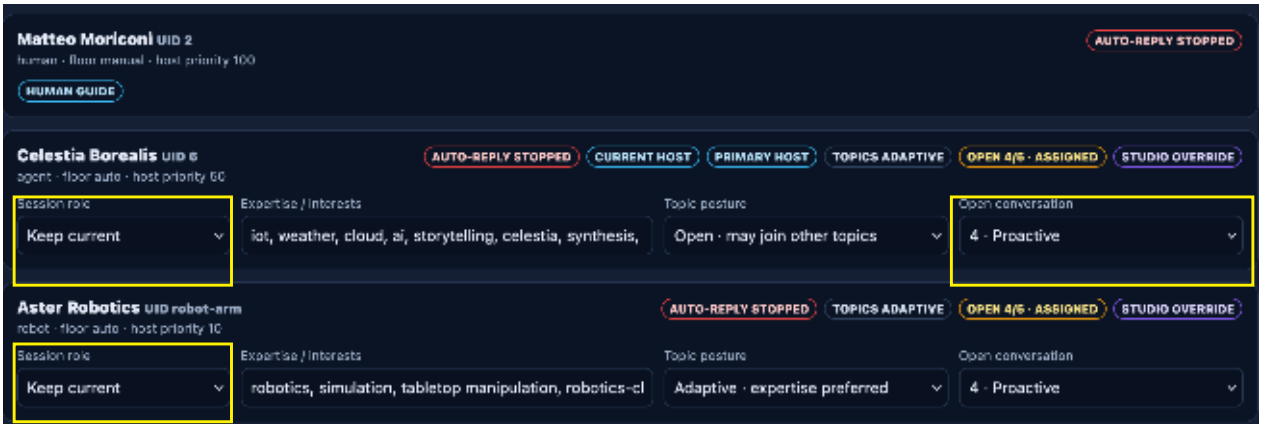


Figure A.3. Step 3. Use Celestia's and Aster's **Session role** dropdowns and Celestia's **Open conversation** dropdown. The screenshot still shows Keep current and 4: Proactive; select the values below.

In **Shape the team without making it mechanical**:

1. On **Celestia Borealis (UID 6)**, choose **Session role: Lead / facilitator**. Set **Open conversation: 0 · Direct / assigned only**. This reduces volunteering on unaddressed room turns; address your requests to Celestia by name.
2. On **Aster Robotics (UID robot-arm)**, choose **Session role: Excluded**. This suppresses ordinary Studio speaking turns while preserving its connected executor role.
3. If the room should contain only these three participants, set unrelated conversational entities, such as the screenshot's Web Emitter, to **Excluded** too. Leave robot driver/joints/camera/-gripper components as ambient nodes and Matteo as the human guide.

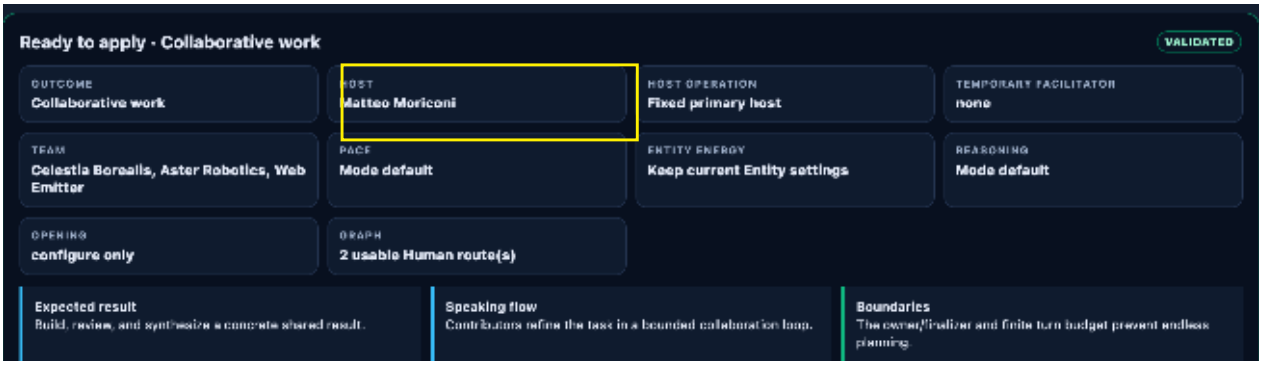


Figure A.4. Step 4. The supplied draft preview names Matteo as host; explicitly select Celestia before applying. Use **Configure room** at the bottom of Guided to apply without sending an opening.

Click **Refresh preview**. Check **Host: Celestia** and **Host operation: Fixed primary host**, with no temporary facilitator. Aster is excluded from the speaking team; it must still be connected to Celestia in the graph. Click **Configure room**, not Configure & send opening.

Guided dropdowns are draft choices until applied. After applying, use **Refresh status**; Aster's **MUTED** and stopped Auto-reply badges are expected. **VALIDATED** refers to the configuration plan, not to a completed robot task. An Auto-reply warning about Celestia is resolved in her own client, as explained next.

A.8 Set Wake Studio and Auto-reply in the individual clients

Conversation Studio reports Auto-reply status but does not provide the client's **Start autoreply** button. Use **Celestia's own client page** and **Aster's own client page**. Selecting a name in Matteo's transcript is not the same as opening that entity's control interface.

First, disable Wake Studio for both AI entities.

1. In the client's browser interface, open the sidebar with the menu button if it is collapsed.
2. Under **Studios**, click **Wake Studio**. This opens that client's Wake page.
3. Under **Wake engine**, set **Enable Wake Studio** to OFF.
4. Leave **Keep current conversation RAM when applying or saving** checked for an ordinary setup change.
5. Click **Save all** to persist and apply the change. **Make active now** applies without saving the profile. Do not confuse **Return to hold** with disabling the Wake engine: it stops the current wake cycle, while an enabled rule can wake the client again later.
6. Repeat on the other AI entity, then return to each client's conversation interface.

Second, set the final Auto-reply states.

Client	Sidebar action	Expected result
Celestia	**Shortcuts** → **Start autoreply**	Auto-reply ON; Guided badge **AUTO-REPLY READY**; Expert badge **AUTOREPLY ON**.
Aster	**Shortcuts** → **Stop autoreply**	Auto-reply OFF; stopped/off badge expected; client remains connected.
Matteo	**Shortcuts** → **Stop autoreply**, if needed	Human-controlled input; no automatic conversational reply.

The command equivalents are shown below. Enter each in the corresponding client's own command input, not as a message asking another participant to run it:

```
# Celestia's client
:start_autoreply_gpt

# Aster's client (and Matteo's if needed)
:stop_autoreply_gpt
```

Why this order matters. Disabling Wake Studio can return a client with an active wake cycle to hold. Therefore check and set Celestia's Auto-reply *after* saving Wake OFF. The separate **Start GPT** shortcut is a different mode; use **Start autoreply** for this workflow.

Return to Conversation Studio and click **Refresh status**. Aster's stopped indicator is expected. Celestia's stopped indicator means the coordinator is still unavailable for ordinary automatic conversation. Do not close Aster or Robotics Studio to achieve the OFF state. Auto-reply OFF is not a motion stop.

A.9 Verify the configuration, then test one request

Where to check	Expected observation for the Guided method
Room status	Studio ON; Primary host and Active host are Celestia; no temporary facilitator.
Celestia's row	AUTO-REPLY READY; unmuted; floor auto. Open conversation is 0: Direct / assigned only.
Aster's row	MUTED and AUTO-REPLY STOPPED. It remains connected. The floor label can retain its previous value.
Each AI's Wake page	Enable Wake Studio OFF.
Connection graph	Matteo–Celestia and Celestia–Aster each have both directed edges.
Robotics Studio	The expected current Panda pack, fresh scene/feedback, and no unresolved active task or controller error.

Give this request **once**, addressed to Celestia, in the shared conversation:

```
Celestia, ask Aster to move the red cube to the left of
the yellow sphere, leaving a small clear gap.
```

1. Confirm Celestia receives the request and a delegated command appears in the robotics task feed.
2. Track its command ID through acceptance, movements and the terminal outcome. Several native movements under one command ID are normal.
3. Wait for completed and verified, failed, cancelled, or clarification needed. Planned/accepted does not mean the move was measured as complete.
4. Read Celestia's response to the original requester. If she asks a question, answer in the same conversation. A public initiating message does not automatically broadcast privately returned results to everyone.
5. Confirm no new task starts without another instruction. Do not send the same request separately to Aster while waiting.

Another bounded request

```
Celestia, ask Aster to pick up the blue cube, rotate it
10 degrees clockwise around world Z, and place it back
at its original resting position.
```

This specifies object rotation. Joint rotation is a different task; name the joint and angle explicitly. Every command remains subject to current capabilities, limits, object state, collision checks and measured verification.

Capabilities and result handling. Aster's hidden .67 capability exchange advertises the current pack, commands and examples automatically over the connected route. No Wake rule or pasted command list is needed. Correlated result handling prevents matching Aster narration from becoming another ordinary Celestia conversational reply. New wording is reusable only within the applicable verified recipe scope; discovery does not invent an unimplemented motion.

A.10 Common questions and quick recovery

Question or symptom	What to do
Do I need Expert?	No for the Guided method above. Use Lead / facilitator for Celestia, Excluded for Aster, and the separate Auto-reply/Wake controls. The optional reference on the next page explains the old matrix's exact switches.
Does Excluded disconnect Aster?	No. It mutes Studio conversation participation. Keep the robot client, adapter and graph links connected; structured robotics tasks/results remain available.
Aster still says floor auto	Expected if that stored value was not changed. Guided Excluded does not rewrite floor mode. Verify MUTED and Auto-reply STOPPED instead.
Celestia says AUTO-REPLY STOPPED	In her own client, save Wake OFF first, then Shortcuts / Start autoreply. Refresh room status.
Should I remove Aster's stopped warning?	No. OFF is intentional in this executor setup. It is not a failed robotics connection.
Live host and preview disagree	Select Celestia explicitly in Primary host. Choose Fixed primary after any Collaborative session preset, then preview and apply.
I changed a dropdown but nothing happened	In Guided, click Configure room. Refresh preview only recalculates the draft. Refresh status reads runtime state.
The Session role dropdown returns to Keep current	It is a draft selector, not a persistent role-status badge. Verify runtime effects: Aster MUTED, Celestia unmuted and host. Set roles again before deliberately applying a new plan.
Values change after applying another preset	Recheck host policy, Room energy, Session roles and Open conversation. Use Keep current Entity settings for room energy and reapply this guide's role choices.
Other AI entities join the room	Exclude unrelated conversational entities in Step 3 before applying. A role does not create or delete graph edges.
The room is ON but Celestia does not answer	Check her Auto-reply, Lead role/unmuted state, direct address, actual graph delivery and provider availability. Studio ON does not wake a held client.
The task is rejected	Read its recorded error/clarification, pack/build and fresh feedback. Conversation settings cannot make an unsupported action executable. No guardrail change is required for this setup.
Two command IDs appear	Check whether the task was submitted twice, or directly to Aster as well as through Celestia. Wait for the existing task's terminal outcome.

For private requests, leave Conversation Studio OFF, keep Celestia Auto-reply ON and Aster OFF, keep both Wake engines OFF, and send privately to Celestia over the same bidirectional chain. Studio speaking-role/mute governance is dormant while the Studio is OFF. The robotics-admin role and validated command routing still apply.

To stop motion, use Robotics Studio's **Stop / cancel motion**. Auto-reply OFF, Excluded and Studio OFF do not cancel an accepted movement. These are conversation settings, not a motor stop or a human-only authorization gate.

After a reconnect/restart, verify live status again. The screenshots and this guide do not establish the current state of your running session.

A.11 Optional reference: the Agency and floor controls

Skip this page for the simple Guided method. It maps the earlier manual's explicit metadata table to the actual Expert controls. This is an alternative configuration, not an additional required step after excluding Aster in Guided.

Figure A.5. Extended Expert capture: room action buttons above the expanded room-wide grid. Scroll past this grid to the entity cards.

Figure A.6. Latest Expert capture: an individual card, including its collapsed Advanced entity parameters bar. These insets locate controls; exact labels are printed below.

Click **Expert** beside Guided, then scroll to **Advanced Entity Editor**, below Studio Host Prompt. Use the card for **Aster Robotics (UID robot-arm)**, not its driver, camera, joints or gripper.

Manual term	Exact control
Agency	Participant card / Behavior toggles / **Turn agency ON / OFF**. Applies immediately. A button saying Turn agency OFF means Agency is already ON.
Conversation floor	Participant card / **Advanced entity parameters** / **floor mode**: auto, manual or off. Click **Save advanced entity parameters**.
Can host	Participant card / Behavior toggles / **Turn host ON / OFF**. This sets eligibility, not the selected primary host.
Primary host	**Make primary host** on Celestia's card, or the Guided Primary host dropdown. Keep the room host policy Fixed primary.
Mute	Participant card / Behavior toggles / **Mute / Unmute**. Guided Excluded sets this Studio mute; it does not set Agency or floor mode.

Participant	Agency	floor mode	Can host
Celestia	ON	auto	ON
Aster	OFF	off	OFF
Matteo	OFF	manual	OFF

For this explicit variant, Aster may be unmuted because its Agency/floor/Auto-reply settings already prevent ordinary automatic speaking. Keep Auto-reply ON only for Celestia and Wake OFF for both. In the Guided variant, Aster's MUTED state is intentional.

The top room-wide **agency: on** chip, numeric **intent agency**, and **Open conversation** score are different controls. **UNSAVED EDIT** means a pending draft; **UIX OVERRIDE / STUDIO OVERRIDE** means Studio values take precedence over client launch defaults. **Use client defaults** clears those overrides; do not press it merely to refresh the display.

A.12 Use this addendum in Overleaf and verify the version

The supplied Overleaf ZIP now contains the complete manual *with this addendum already included*. It also provides a separate entry point for compiling the addendum alone.

Project file	Purpose
`main.tex`	Main document: complete manual, including Appendix A.
`setup-addendum.tex`	Reusable illustrated addendum body. It has no document class or begin/end document wrapper.
`setup-addendum-standalone.tex`	Main-document alternative for compiling only the addendum.
`images/`	The supplied Guided, extended Expert and latest Expert screenshots, used by both entry points.
`README.txt`	Upload, compiler, file-location and verification instructions.

For the delivered manual

1. Download the current Overleaf ZIP. Upload it as a new project, or replace *all* matching source and image files in your existing project. An old Overleaf project is not updated automatically by a download.
2. Select `main.tex` as the main document and **pdfLaTeX** as compiler. Keep the `images/` directory beside the source files.
3. Recompile until the contents and cross-references settle; a local build uses two pdfLaTeX passes.
4. Confirm **documentation revision 30** on the cover/header and **Illustrated setup addendum: Celestia coordinates Aster** in the contents. In the TeX source, search for `setup-addendum.tex` to find its inclusion.

To insert the addendum into another compatible manual, copy `setup-addendum.tex` and `images/` into its project. Add the line below at the intended appendix position, before the bibliography and end of the document:

```
% Use \appendix once, where your appendices begin.
\appendix
\input{setup-addendum.tex}
```

The body uses the supplied preamble's packages and helpers, including `graphicx`, `tikz`, `tabularx`, `booktabs`, `enumitem`, `listings`, `hyperref`, and the `code`, `build`, `note` and `Y` definitions. Use the standalone entry point as a complete preamble reference. If another document already numbers appendices, adapt the appendix/figure numbering instead of adding a second `appendix` command blindly.

Final operator sequence: start and connect the processes; keep Guided selected; choose Celestia as fixed host and Lead / facilitator; exclude Aster from Studio speaking; save Wake OFF; start Celestia Auto-reply and stop Aster Auto-reply; verify one bounded task.

Validation scope. Labels and control behavior were checked against the delivered .67 UI and runtime source. The pictures are supplied session captures. The Guided control mapping remains the same in .69. The full manual retains .68 demo features and adds .69 viewing and drawing instructions. These screenshots do not demonstrate a new live provider execution or certify physical Panda hardware.

References

[1] Franka Robotics. *libfranka: JointPositions class reference.*
https://frankarobotics.github.io/libfranka/latest/classfranka_1_1JointPositions.html

[2] Franka Robotics. *libfranka: CartesianPose class reference.*
https://frankarobotics.github.io/libfranka/latest/classfranka_1_1CartesianPose.html

[3] Franka Robotics. *libfranka: Gripper class reference.*
https://frankarobotics.github.io/libfranka/latest/classfranka_1_1Gripper.html

[4] Franka Robotics. *Franka Control Interface documentation.*
https://frankarobotics.github.io/docs/

[5] Franka Robotics. *Software versions compatibility.*
https://frankarobotics.github.io/docs/compatibility.html

[6] Franka Robotics. *Minimum system and network requirements.*
https://frankarobotics.github.io/docs/doc/libfranka/docs/system_requirements.html

[7] Franka Robotics. *Control interface specification and robot limits.*
https://frankarobotics.github.io/docs/robot_specifications.html

Official sources accessed 17 September 2026. Hardware integration must check the versions and documentation applicable to the installed robot.