ORBITUS ROBOTICS / WHITE PAPER

A recovery-first CAN takeover plan for an experimental robot

From offline evidence to a reversible, exclusive controller handoff

OR-WP-2026-003 | Version 1.1 | November 4, 2026

Orbitus Robotics · Project lead: Rodolfo Aramayo

Abstract

Offline motion software can pass extensive tests while its real-machine handoff remains unproven. This paper describes the recovery and commissioning boundary for replacing ROB's Amber host controller. The recorded candidate passed 84 Linux checks in Release and Address/Undefined Behavior Sanitizer configurations, 58 portable C++ groups under Thread Sanitizer, and five inert rehearsals on the target host with zero network syscalls. Physical transmission remained unavailable. We distinguish verified file archives from bootable recovery, and logical single-writer design from physical CAN exclusivity. The proposed handoff combines supported inactive hardware, legacy-writer exclusion, adapter queue validation, namespace isolation and a fresh generation with bounded feedback. A November 4 follow-up closes an earlier protected-file backup gap, while whole-disk imaging, spare boot restoration and physical stopping remain open. The contribution is a traceable acceptance framework, not a completed takeover.



RESEARCH RECORD

An evidence-based plan for backups, exclusive CAN ownership, fresh controller generations and measured stopping before physical streaming.

Evidence originates in the 2026-11-04 report. Technical white paper · Not peer reviewed.

ROB during the October 29, 2026 can session. Still from Kierie's video. A loaded arm illustrates why recovery must consider gravity and retained targets; no CAN takeover is shown.

Keywords: CAN ownership, robot commissioning, recovery, backup, fault containment

ROB: history, purpose and software

ROB (R.O.B.) is Orbitus Robotics' experimental tracked humanoid: a platform for robotics education, human-robot interaction and supervised manipulation. Independent treads, a rotating torso, two seven-joint arms and grippers, cameras and range sensing connect software to a physical machine. Can-handling practice, stage demonstrations and spatial teleoperation give the project concrete tasks to investigate. [8,9,10]

The project's archive reconstruction records ROB-specific Mac serial-control work in 2017-2018 and a surviving Cerebro Git root on January 1, 2018. Later histories add speech, perception experiments and controller arbitration. Public galleries document builds and appearances from 2019 onward. The fresh Cerebro repository of August 5, 2025 imported an existing v5 application; it was not the project's beginning. The pre-v5 originals remain local, with provenance summarized in the supporting record. [8,10]

Codebase / platform	Role in ROB
Cerebro / macOS	Swift, Objective-C/Objective-C++, C++ and Python helpers coordinate perception, dialogue, command admission and robot I/O.
ROBController / iPhone, iPad, Watch	Human command, maps, telemetry and explicit action approval.
ROBControllerVision / visionOS	SwiftUI spatial controls, live video and time-limited arm authority.
ROBArduino + RPLidar project	Base firmware handles low-level drive; the lidar application supplies scans and map observations.
Amber host + ContinuousArm	Existing Ubuntu arm runtime; the C++ replacement owner and codec live in Cerebro/Tools/ContinuousArm.

The public source index links six repositories and suggests starting files. The architecture separates human intent and AI-assisted planning from motor-facing ownership. This paper examines that latter boundary. It does not establish that every project feature is active together, or that the replacement controller has been physically commissioned. [8,9]

1. Readiness is a set of separate claims

The original eight-page November 4 plan marked the candidate as ready for backup and read-only preparation. It did not mark privileged takeover, adapter reset, physical transmission or motor-stop commissioning complete. The deployed state matters as much as the source state: staged files and passing tests are not installed protection. [1]

Evidence as recorded	Result	Boundary
Linux validation	84 checks in Release and ASan/UBSan	53 portable + 8 transport + 11 runtime + 5 rehearsal + 7 preflight
Thread Sanitizer	58 portable C++ groups	53 core groups + 5 rehearsal groups
Inert target-host rehearsal	5 cases; zero network syscalls under strace	No CAN transmission
Recovery/startup source	27 offline tests	Protected guards not installed
Candidate mode	Physical transmit unavailable; not installed; service not enabled	No physical streaming readiness

Controller source is d98ebd3791b9509249212e2c754f4061f78458f9; the staged package hash is 3628a1d4bc60aeacc40fb9fc3930ebb65419a019f3afe730c5734853fd9a909e. These identities bind the results to a particular candidate. They are not approval receipts or evidence about another build. [1,2]

> November 4 addendum: the revision-1 report's unreadable protected-policy gap was later closed by a read-only backup of three regular policy files, with hashes and ownership/mode metadata verified. The source policy was unchanged. Full-disk imaging, full system-state inventory and a spare-disk boot test were still unfinished. [1]

2. Prove recovery before replacing ownership

The verified original-controller archive contains 48 payload files totaling 43,686,735 bytes: both host cores, 42 mapped libraries, launch files, gateway code and startup material. That archive is valuable for file restoration. It does not contain the entire boot disk, prove a bootable restore, or back up actuator EEPROM. Seven additional readable startup/recovery files were captured before the protected-policy follow-up. [1]



Proposed recovery gates - completion requires recorded evidence at each step.

Figure 1. Proposed recovery chain. Image verification, an independent copy and an observed spare boot are distinct acceptance gates. None is inferred from file checksums alone.

The observed boot disk was approximately 64.02 GB, with EFI, a separate boot partition and an LVM-backed root filesystem. Recovery needs the partition table, boot structures, LVM metadata and root data, with UEFI settings captured separately. The plan recommends imaging the unmounted original, restoring to an identified compatible spare, and working on that clone while preserving the original disconnected. Clonezilla documents whole-disk, LVM2 and UEFI workflows, with the imaged filesystem unmounted. [4]

Gate	Acceptance evidence
B1: identify media	Record exact source/destination identities, capacity and partition layout; recheck in recovery environment.
B2: capture image	Verify a complete offline whole-disk image, not only a home directory or selected files.
B3: independent copy	Verify a second copy on different physical storage.
B4: restore and boot	Restore to a spare and verify boot/configuration with motor power off and CAN disconnected.

A destination must be selected by verified identity and capacity, not a remembered device name. The spare must be physically compatible and large enough for the selected restore method. Boot success is still separate from correct actuator calibration or mechanical recovery.

3. Exclude old writers and old traffic

A single-threaded new controller does not become the only motor writer merely by starting. SocketCAN allows multiple sockets to send on the same interface. The handoff must exclude legacy processes, restart routes and direct serial access as well as constrain the replacement. Namespace lifetime also matters: physical interfaces return to the initial namespace when their network namespace is freed. The proposed ownership therefore uses a pinned, root-managed namespace with controlled driver admission. [3,5,7]

Handoff gate	Proposed completion condition
T1: supported start	Both arms mechanically supported and drives disabled; all control sessions coordinated.
T2: legacy exclusion	Exact prior service/protected-file state recorded; guards and persistent inhibit verified; legacy writers gone.
T3: queue/adapter behavior	Old host queues closed; adapter reset/reinitialization and residual-traffic behavior measured with motors disabled.
T4: physical CAN isolation	Correct real slcan interface/arm identity and driver-only access verified; virtual CAN alone is insufficient.
T5: fresh generation	Cached targets discarded; new per-joint state, mode, limits and stationary history meet a reviewed start grant.

The advertised CANable2 source gave a reason to investigate adapter queues, but it was not a verified image of the flashed firmware. Closing the CAN interface cannot be treated as a proven adapter queue flush. Reset behavior, serial buffering and the age of commands reaching the wire require supported hardware evidence. [6]

ROB's physical RIGHT mapping is CAN10/L-10; LEFT is CAN11/R-11. These labels must be checked against actual adapter and joint identity. Startup/recovery couples both arms, so a one-arm test does not by itself make a partially controlled two-arm handoff acceptable.

> A failed handoff leaves output disabled and preserves the last verified state. Crash or reconnect must not replay cached targets, automatically remove the inhibit, or revive the legacy core.

4. Measure stopping before continuous targets

A bounded software brake depends on the process and transport remaining able to publish. A lost process, bus or power supply may defeat that path. Before physical streaming, the project still needs measured stop latency, travel, current, mode and gravity behavior, including evidence about any drive-side timeout. A hardware shutdown switch supports supervised commissioning; it does not establish an automatic firmware watchdog.

• C0: verify physical joint IDs, signs, zeros and modes; establish measured position/rate/current limits, transport-age bounds, whole-arm sweep and stop reserve.

• M1: after an explicit operator resume, perform one reviewed, low-speed, empty-arm raise followed by measured Stop. Choose the joint and sign from fresh posture, not a synthetic test angle.

• M2: after the first motion and stop gates pass, apply a finite set of nearby mid-motion target changes through the same persistent owner. Record accepted/applied sequences and bounded tracking.

• End a run on feedback loss, unexpected current, expired targets or ownership loss. Preserve the evidence and require a fresh reviewed start rather than automatic replay.

The execution record should contain the source/package identities, arm and operator, initial measured joint state, per-joint timestamps, commanded state, current/mode, generation, accepted/applied sequence numbers, Stop outcome and each gate's evidence reference. All original execution gates were Pending; no physical Pass is assigned in this paper.

Headless arm-only tests can be planned, but an absent head camera cannot certify the world geometry. Face pointing and VR require their own mapping and tracking-loss validation. A software-held state, quiet receive counters or stale cached telemetry does not establish physical stationarity.

> The next useful result is a measured, supported stopping experiment with a known controller and transport state. More offline check counts cannot answer that hardware question.

5. Rollback, reusable lessons and publication record

Failure class	Recovery principle
Only inactive staging changed	Retain the original runtime; preserve logs and identify task-owned staging precisely.
Handoff or feedback gate fails	Keep output disabled and ownership inhibits in place; do not retry uncertain buffered commands.
Return to vendor control	Support and disable arms, exclude the replacement writer, resolve queues, restore exact changed files/service state, then separately authorize motion.
Clone fails to boot	Use local console and the preserved original or verified image with motor power off and CAN disconnected.
Unexpected motion	Use the independent shutdown path, account for gravity/support, end the run and investigate.

The reusable lesson is that recovery and ownership should be tested before live control is expanded. A file hash proves identity, a simulated plant tests software under chosen assumptions, and a boot rehearsal tests restoration. Each answers a different question. A defensible release record carries those boundaries forward instead of collapsing them into a single "ready" label.

This edition adapts the November 4 revision-1 report and incorporates the later protected-policy backup correction recorded in the pinned handoff. The machine-readable supplement records the evidence dates and open gates. Local credentials, private backup paths, disk serials and operator conversation are excluded. The plan is a case study, not a universal robot commissioning procedure.

No new robot command, reset, installation, reboot or physical trial was performed to prepare this publication. The companion papers explain the trajectory handoff and host-boundary findings that motivated this plan.

References and supporting records

[1] November 3-4 handoff, backup correction and open gates (f5861f8)

https://github.com/RudyAramayo/Cerebro/blob/f5861f82e9025c2df2eb71ab93c74c1695ba5f6d/docs/calibration/amber-overnight-preparation-2026-11-03.md

[2] Source-bound overnight manifests and verification

https://github.com/RudyAramayo/Cerebro/tree/f5861f82e9025c2df2eb71ab93c74c1695ba5f6d/docs/calibration/evidence/2026-11-03-amber-overnight

[3] Recovery/startup ownership source (7c3b244)

https://github.com/RudyAramayo/Amber-HomeFolder/blob/7c3b2449936a017c5f7049f6d562764a9fdb3f94/docs/amber-stack-recovery.md

[4] Clonezilla: supported imaging and recovery capabilities

https://clonezilla.org/

[5] Linux network namespace lifetime

https://man7.org/linux/man-pages/man7/network_namespaces.7.html

[6] Advertised CANable2 source; flashed image not verified

https://github.com/normaldotcom/canable2-fw/blob/b158aa7ab106bc018715cd4d5f02f544f9de1830/src/can.c

[7] Linux SocketCAN: multiple senders

https://docs.kernel.org/networking/can.html

[8] ROB project chronology and public source index (v1.1)

https://www.orbitusrobotics.com/whitepapers/rob-project-background.json

[9] ROB software platform and operator interfaces

https://www.orbitusrobotics.com/products/

[10] ROB development and Maker Faire photo archive

https://www.orbitusrobotics.com/makerfaire/

Web edition and supplements: https://www.orbitusrobotics.com/whitepapers/amber-can-takeover-and-recovery-plan/

Version 1.1 adds ROB history, purpose, functionality and a software-stack overview. Recorded test results and physical commissioning status are unchanged.

Prepared with AI assistance from project reports, source code and recorded test evidence. Technical claims are tied to the cited records. No new physical robot trial was performed for this publication.

Photography: Kierie, October 29, 2026. Diagrams: Orbitus Robotics. Correspondence: orbitus@orbitusrobotics.com