

SETUP B · VIRTUAL PRODUCTION · HAMBURG

THREE TAIL 2, ONE LIVE UNREAL SET

How three OBSBOT Tail 2 drive a live-keyed Unreal Engine set – one Ethernet cable per camera, no lighthouses, no tracker calibration per shoot.

3×

Tail 2 in a 5 × 5 × 2.9 m green-screen set

60 Hz

FreeD tracking, two samples per 1080p30 frame

10 min

to bring the whole system up

THE CHANGE

From optical tracking to in-camera FreeD

Before: HTC Mars + lighthouses

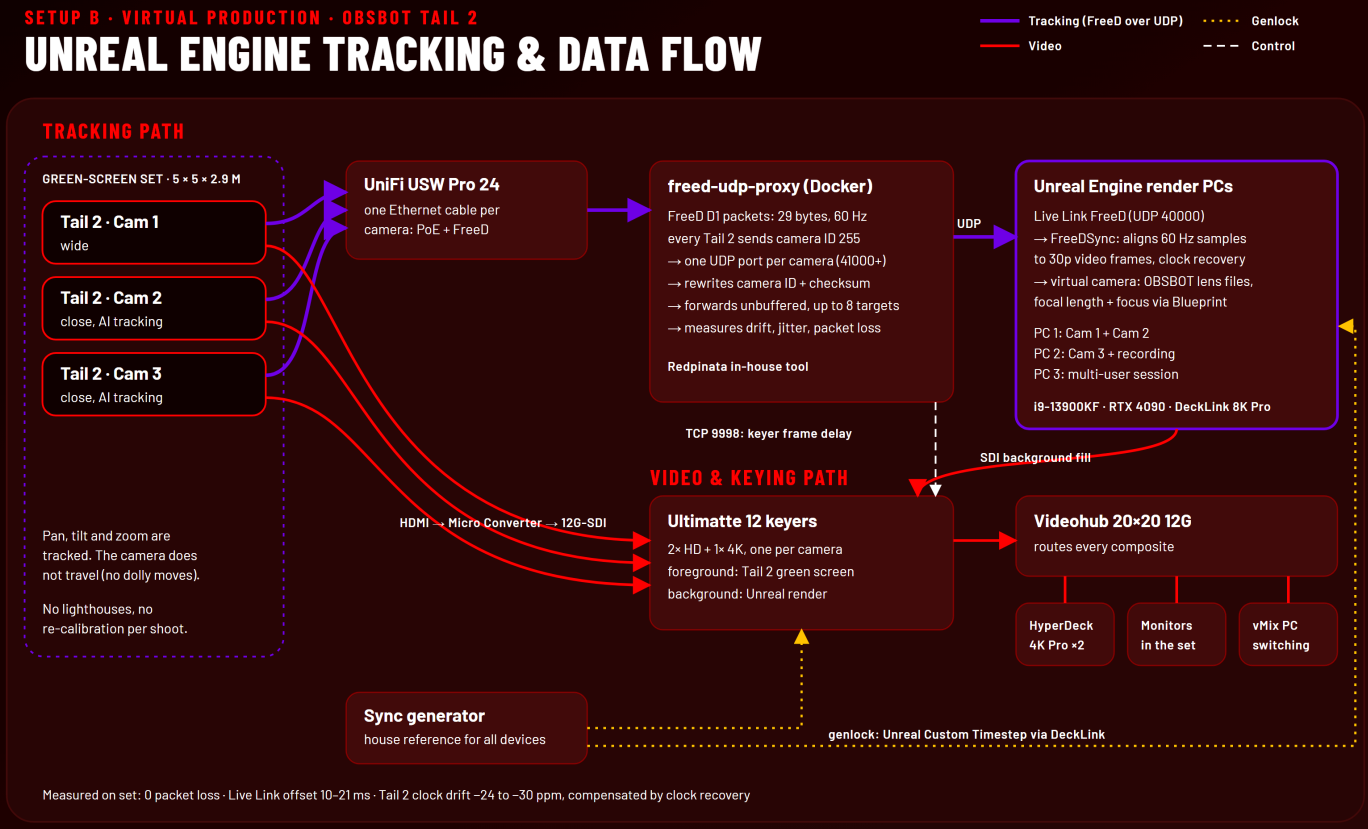
Optical tracking fought the lights: cover a lighthouse and tracking dropped; keep the lighthouses clear and the lighting suffered. Every production started with a full calibration: trackers on marked points, reset, measure.

With Tail 2

Each camera reports its own pan, tilt and zoom over Ethernet. Positions in the set are known and stay put. Three identical cameras share the same lens, so one set of lens files applies to all of them.

SETUP B · VIRTUAL PRODUCTION · OBSBOT TAIL 2

UNREAL ENGINE TRACKING & DATA FLOW



Measured on set: 0 packet loss · Live Link offset 10–21 ms · Tail 2 clock drift −24 to −30 ppm, compensated by clock recovery

The camera can pan, tilt and zoom. It cannot travel. We mitigate that by rotating toward the talent: the host can stand up and move, and the shot stays with them. Without lighthouses, large area lights and a monitor inside the set are possible again.



OBSBOT Tail 2, status light on.



Close-up camera in its fixed position, hosts in the background.

What runs where

Layer	Component	Role
Cameras	3× OBSBOT Tail 2	Picture plus FreeD tracking (pan, tilt, zoom, focus): one wide, two close-ups
Network	Ubiquiti UniFi USW Pro 24 (control room)	One Ethernet cable per camera: PoE power, VISCA over IP, FreeD
Video out	HDMI from each Tail 2, Blackmagic Micro Converter HDMI to SDI 12G	HDMI runs to the control room are already in place; the converter turns them into SDI for the keyer
Tracking relay	freed-udp-proxy (Redpinata, Rust, Docker)	Gives every camera its own ID, measures timing, talks to the keyers
Rendering	2× Unreal Engine PC (i9-13900KF, RTX 4090, DeckLink 8K Pro)	PC 1 renders cameras 1 + 2, PC 2 camera 3 and records
Session	Third PC	Unreal multi-user session for all render PCs
Keying	3× Ultimatte 12 (2× HD, 1× 4K)	One per camera, live composite of camera and Unreal background
Routing	Blackmagic Videohub 20×20 12G	Composites to recorders, monitors and vMix
Recording, switching	2× HyperDeck Studio 4K Pro, vMix PC (DeckLink Quad 2)	ISO recordings and the live program
Sync	Blackmagic Mini Converter Sync Generator	House reference for keyers and DeckLink cards
Control	OBSBOT remote, joystick controller, OBSBOT app, Companion	Camera moves, presets, tracking on/off, FreeDSync commands



The green-screen set: cyclorama, studio lights, a Tail 2 on its tripod.



Hosts on set, area lights and the on-set preview monitor behind them.



Control room: vMix, audio and the multiview of the keyed composites.



Lounge monitor showing the keyed Tail 2 composites.

One cable: power and FreeD

FreeD D1 packet (29 bytes)

Byte	Field	Encoding
0	Message type	0xD1
1	Camera ID	255 on every Tail 2
2–10	Pan, tilt, roll	24-bit, ÷ 32768 = degrees
11–19	X, Y, Z	24-bit, ÷ 64 = mm
20–22	Zoom	24-bit raw
23–25	Focus	24-bit raw
26–27	Spare	–
28	Checksum	0x40 – sum of bytes 0–27

60 packets per second per camera, on 1080p30 video: two samples per frame.



Tail 2 in the set: one Ethernet cable with PoE carries power, control and FreeD.

The ID problem

Every Tail 2 sends camera ID 255. Unreal Live Link FreeD then sees three cameras as one. That is why we built the proxy.

freed-udp-proxy – Redpinata in-house tool

A Rust service in Docker on the host network. One UDP port and one thread per camera. Each packet is checked (length, 0xD1), gets its new camera ID and a recalculated checksum, and goes straight on to Unreal – no buffer, no queue, up to eight targets. A web GUI manages the routes, shows a live packet log and a timing panel for drift, jitter and delay.

Port	Protocol	Use
41000, 41001, …	UDP	FreeD in from each camera, one port per route
40000	UDP	FreeD out to Unreal Live Link (same port, distinct IDs)
41080	TCP / HTTP	Proxy web GUI and API
9998	TCP	Ultimatte Ethernet Protocol: frame delay, keyer presets

Why a free-running camera drifts – and how FreeDSync follows

The Tail 2 has no genlock input. Its clock runs free and differs from the house reference by a few ppm – we measured –24, –30 and –28 ppm on our three cameras. The Ultimatte 12 frame-syncs the picture to the reference, so every few minutes it repeats or drops one frame. A fixed tracking delay only fits part of that cycle; for the rest, the background is one frame off.

Slip cycle

$$\text{cycle} = 1 / (\text{fps} \times |\text{ppm}| \times 10^{-6})$$

Measured drift	Slip every (30p)
–24 ppm	≈ 23 min
–28 ppm	≈ 20 min
–30 ppm	≈ 19 min

FreeDSync – Redpinata Unreal plugin

- A Live Link interpolation processor ("FreeD Frame Sync"), UE 5.6 and 5.8.
- Recovers the camera clock from raw packet arrival times: packets can only arrive late, never early, so the lower envelope is the clock. Accuracy about 0.15 ms.
- Recovers the genlock clock from the engine ticks.
- On every tick it picks the FreeD sample that belongs to the frame the keyer is showing. When the keyer slips, tracking slips with it.

Mode	Behaviour
Observe	Model runs and logs predicted slips; normal interpolation
Mirror	Follows the keyer's frame slips sample by sample
FixedLatency	Fixed phase offset from the genlock tick

Proxy timing gate	Limit
Measurement window	≥ 10 min
Jitter (envelope RMS)	≤ 1 ms
Delay p99	≤ 50 ms
ppm spread	≤ 2 ppm

0.15 ms
clock reconstruction accuracy

10–21 ms
Live Link time offset – excluded from the clock fit

0
FreeD packets lost on set

Safe fallback

Until the clock model is stable – about 5–10 seconds after a start, after an outage of more than 10 seconds or a camera restart – FreeDSync falls back to normal Live Link interpolation and writes a warning to the log. Shorter gaps do not reset the measurement.

Controlled from Companion

Every camera has a label (CAM1–3). Status, mode, MarkSlip, Nudge, Trace and Reset run as console commands or as Companion buttons through the Unreal Remote Control API – on every render PC that uses the camera.

VIDEO & KEYING PATH

Green in front, Unreal behind

Foreground

The Tail 2 has HDMI and SDI outputs. We use HDMI because those cables already run to our control room; a Micro Converter HDMI to SDI 12G feeds each camera into its own Ultimatte 12.

Background

The Unreal PCs render one world per camera and send it through DeckLink 8K Pro cards as SDI fill into the keyers.

After the keyer

The Videohub routes every composite to the two HyperDecks, the monitors in the set and the vMix control-room PC.

Sync and control

The sync generator references keyers and DeckLink cards; Unreal follows via Custom Timestep. The proxy reads and sets each keyer's frame delay (0–14 frames) over TCP 9998.



Raw Tail 2 feeds before keying: wide shot and close-ups on green.



vMix: keyed composites next to the raw green-screen feed.



Control room: vMix with the keyed composites, Videohub and HyperDecks below.



Tail 2 in front of the control-room monitors.

Set up once, switch on in ten minutes

Once per studio

1. **Height**
   Measure floor to lens for each camera and enter it in Unreal.
2. **Position**
   Place the cameras in a model of the studio in Unreal. The physical positions stay fixed.
3. **Lens**
   Load the OBSBOT lens files into Live Link. A Blueprint reads focal length and focus and drives the virtual camera.
4. **Live Link**
   FreeD source in Engine Time mode, engine genlocked. Per subject: interpolation "FreeD Frame Sync", label CAM1–3, 30 fps.

Timing, per camera

1. **Measure**
   At least 10 minutes in the proxy timing panel until all gates are green.
2. **Observe**
   Wait until the drift uncertainty is within ±0.5 ppm (about 6 min) and the predicted slips match the visible ones.
3. **Lock**
   Set the base delay right after a slip, switch to Mirror, mark the next visible slip from a Companion button.
4. **Fine-tune**
   Nudge by ±1 frame or sample until a pan shows no offset, then save the Live Link preset.

Every production day

Switch on, start the Unreal instances, pick the world, go. That is ten minutes for one person. Keying takes about 30 minutes at most: 80% of the quality is there after ten minutes; the last 20% is light and keying craft, 0.5–1.5 hours.

10 min
system up, one person

~30 min
keying set-up at most

0
tracker calibrations per production



Operator steering a Tail 2 with the OBSBOT remote.



Control room: the keyed composites on the multiview above the vMix station.

What the composite looks like



Wide composite: Tail 2 keyed live in front of the Unreal environment (HyperDeck recording, 21 Oct 2026, 1080p30).



Tracked close-up with a parallax-correct background.



Second close-up, straight from the keyer.

Limits we know about

- The camera rotates and zooms but does not travel. We rotate toward the talent.
- A lost FreeD packet is detected only at the end of a 30-sample block. Measured on set: no losses.
- If a sample is late because of network congestion, FreeDSync uses the newest one available (p99 delay 8–40 ms).
- The keyers must run on the analog reference so the frame sync hits the camera picture – checked on all three.
- Light and keying stay craft. Light presets per Unreal world via DMX are in progress.