Kellogg movement measurements

Findings

The recording supports **release-dependent jump heights with intermediate values**, not just two fixed jump heights. It also supports **constant horizontal acceleration toward a target speed**, including braking and reversing.

One shared discrete model reproduces **all 15 complete jumps / 821 airborne position samples exactly**, after fitting only a subpixel starting phase and, for short jumps, a candidate release frame for each occurrence. The same horizontal constants reproduce **17 of 20 transition segments / 513 samples exactly**. Segment samples can overlap. Three horizontal exceptions are retained and discussed below.

“Exact” means equality of the observed integer full-sprite-box coordinates. It does not mean we have recovered the original source code, collision box, keyboard polling order, or every internal state. Input events were not recorded. Release and direction-change times below are inferred model events, not measured keys.

Recording and coordinates

- Source recording: `jump and run.avi`
- SHA-256: `4a1d3dca7986379f0acfcab5b84afff8b875078ceb339788a32f2fec164fcf1f`
- 2017 frames; 70.086304 recorded frames/s; original rational PTS retained.
- 1890 frames have accepted camera and player world positions. Accepted motion ends before the large dialogue overlay; unknown frames are not filled.
- The source is 640×400; analysis uses 320×200 logical pixels with a 320×176 gameplay viewport. **Multiply distances by two for original AVI pixels.**
- Player measurement is the complete **32×56 sprite canvas**, including transparent padding, matched against extracted animation frames. It does not follow feet.
- Camera is registered independently against the original map. World box position is camera position plus viewport box position. The constant floor baseline is world box top **Y=584**, box bottom **Y=640**.

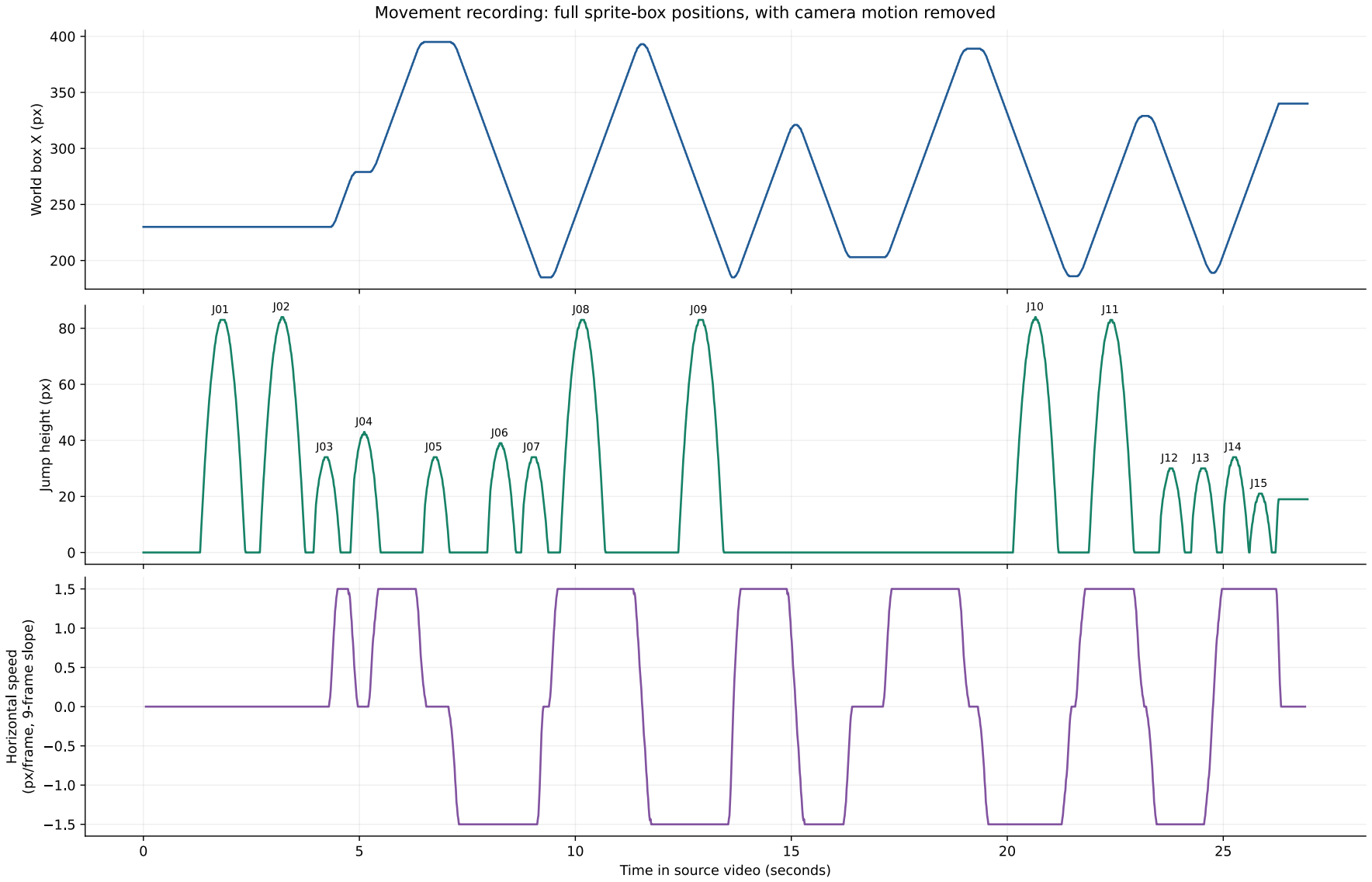

Figure 1: Recorded movement

Shared candidate constants

Here “frame” means one recorded-frame interval in this experiment. The matching one-step model strongly supports one movement update per captured frame during these sequences. Its scheduling on other emulator settings is not established.

Parameter	Logical units per frame	Equivalent using this recording’s time base
Maximum horizontal speed	1.5 px/frame	105.129456 px/s
Horizontal acceleration / braking magnitude	0.125 px/frame²	614.011251 px/s²
Initial upward jump speed	4.5 px/frame	315.388368 px/s
Downward gravity	0.125 px/frame²	614.011251 px/s²
Upward-speed limit after early release	2 px/frame	140.172608 px/s
Maximum falling speed	4 px/frame	280.345216 px/s

All six values can be represented in eighth-pixel units: **12, 1, 36, 1, 16, 32**. That makes an integer/fixed-point implementation plausible; it is not proof of the original numeric representation. Do not round the measured time base to 70 and then silently mix per-second and per-frame parameters.

Jumps: height depends on release time

The measured peak heights are **21px (1×), 30px (2×), 34px (4×), 39px (1×), 43px (1×), 83px (4×), 84px (2×)**. Six full jumps last 74 frame intervals (1.055841s), reaching 83 or 84 pixels. The one-pixel difference is reproduced by the subpixel launch phase, with unchanged physics constants. Short jumps reach several distinct intermediate heights.

Before release and before the falling-speed limit, with downward-positive Y:

```
Y(n) - Y(0) = -4.5*n + (0.125/2)*n*(n-1)
            = -4.5625*n + 0.0625*n²
```

If the speed-limit event occurs after r full ascent steps, the next step uses `max(-4.5 + 0.125*r, -2)` as vertical speed. The resulting trajectory is two parabolic pieces with the same gravity and a sudden upward-speed reduction. It is not best explained by selecting a smaller launch velocity from the start: short jumps initially share the full-jump ascent.

For example, J03 and J05 follow **identical 34px trajectories** and both fit a release event after four full ascent steps. J04 reaches 43px with six full steps; J06 reaches 39px with five; J12/J13 reach 30px with three; J15 reaches 21px with one. These are inferred full-step counts, **not exact physical button hold durations**. Keyboard polling, the unobserved press time and frame phase can shift that mapping.

The final descending samples of full jumps reveal a **4px/frame fall-speed cap**. Without it, the model is too far down by up to two pixels near landing. Adding this one shared cap resolves those errors across all six full jumps.

Jump	Source frames, inclusive	Start (s)	Height (px)	Duration (frames)	Inferred full-speed ascent steps before release	Subpixel phase	Model RMSE (px)
jump-01	92–166	1.3127	83	74	not required	0.250	0.000
jump-02	189–263	2.6967	84	74	not required	0.000	0.000
jump-03	276–320	3.9380	34	44	4	0.750	0.000
jump-04	336–385	4.7941	43	49	6	0.000	0.000
jump-05	453–497	6.4635	34	44	4	0.750	0.000
jump-06	558–605	7.9616	39	47	5	0.000	0.000
jump-07	613–657	8.7464	34	44	4	0.375	0.000
jump-08	676–750	9.6453	83	74	not required	0.625	0.000
jump-09	868–942	12.3847	83	74	not required	0.375	0.000
jump-10	1411–1485	20.1323	84	74	not required	0.125	0.000
jump-11	1534–1608	21.8873	83	74	not required	0.875	0.000
jump-12	1648–1690	23.5139	30	42	3	0.625	0.000
jump-13	1700–1742	24.2558	30	42	3	0.125	0.000
jump-14	1750–1794	24.9692	34	44	4	0.625	0.000
jump-15	1795–1831	25.6113	21	36	1	0.875	0.000

“Not required” means no release cap is needed to explain the trajectory. A release after upward speed has already fallen below 2px/frame would have no visible effect. The video cannot distinguish that from holding jump throughout.

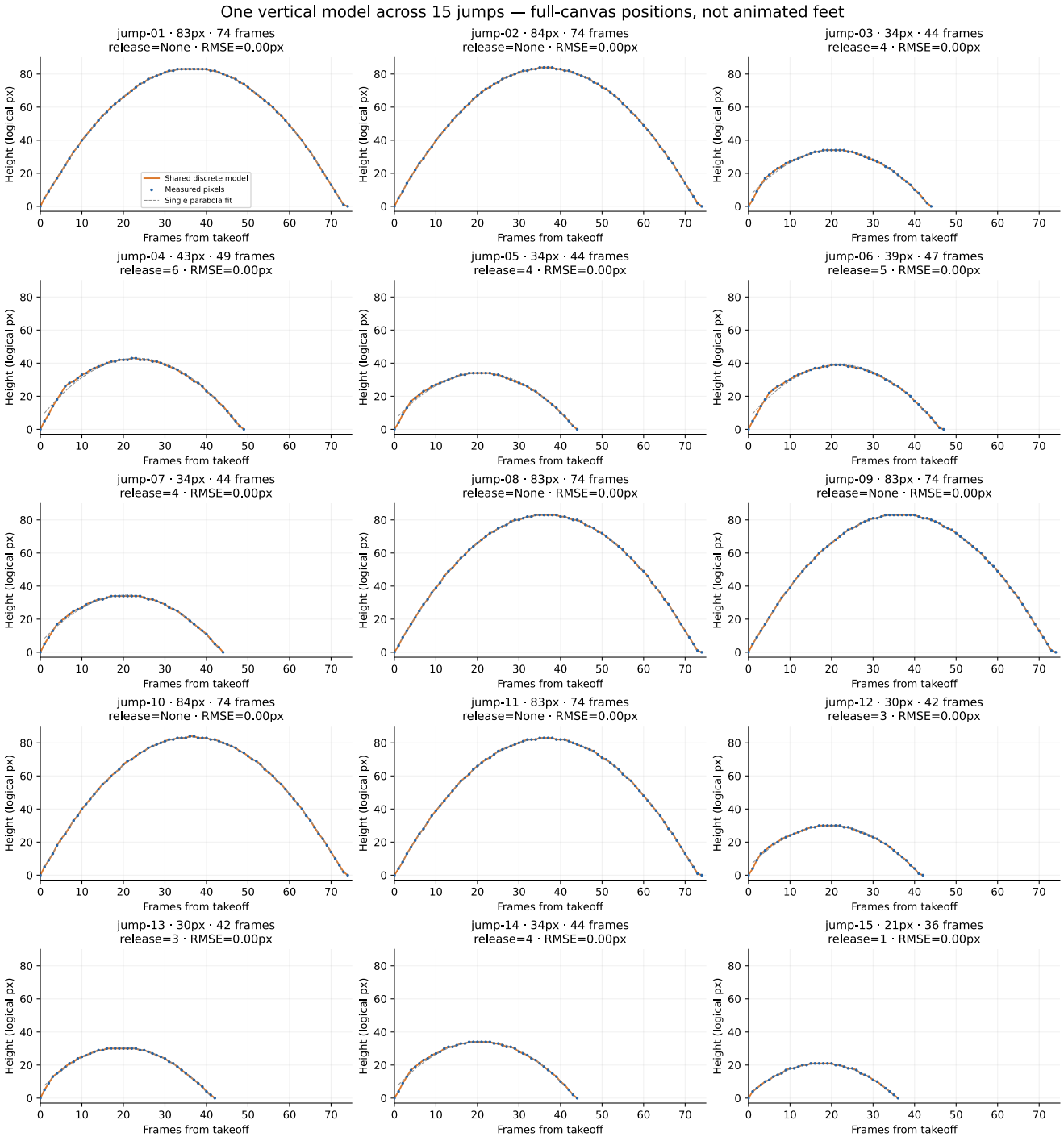

Figure 2: Individual jump fits

Horizontal movement, stopping and reversal

The candidate rule is a ramp toward a target velocity:

```
target_vx = direction * 1.5       # -1 left, 0 neutral, +1 right
vx = approach(vx, target_vx, 0.125)
x += vx
```

This takes **12 updates** to reach full speed from rest and 12 to brake from full speed to rest; a direct full-speed reversal takes **24 updates** to reach full speed in the opposite direction. With the update order shown, stopping after release travels **8.25 further pixels** before the velocity reaches zero. A reversal first travels the same distance in the old direction. An alternative placement of the input event relative to the frame changes the reported first/last sample, so these distances should not be confused with a host-keypress-to-stop measurement.

The horizontal position function during an uncapped ramp is quadratic; when the speed reaches ±1.5 it becomes linear. The recorded 1px/2px alternating steps are consistent with that 1.5px/frame speed and integer rendering. Using those steps as instantaneous acceleration would create false oscillations. Fits use positions; smoothed slopes in the overview/acceleration plots are for visualization only.

Repeated stops followed by a brief standstill were split into two events rather than incorrectly fitted as a slower, single reversal. Matching events occur both on the ground and in the air, consistent with horizontal control during jumps. The video does not establish all possible simultaneous-key policies.

Segment	Source frames	Type	Velocity (px/frame)	Inferred change frame	Model RMSE (px)	Exact position match
horizontal-01	296–321	start	+0 → +1.5	303	0.000	yes
horizontal-02	327–354	stop	+1.5 → +0	336	0.000	yes
horizontal-03	360–387	start	+0 → +1.5	368	0.000	yes
horizontal-04	437–465	stop	+1.5 → +0	445	0.000	yes
horizontal-05	490–518	start	+0 → -1.5	500	0.000	yes
horizontal-06	634–655	stop	−1.5 → +0	640	0.477	no — exception
horizontal-07	653–678	start	+0 → +1.5	660	0.000	yes
horizontal-08	790–830	reverse	+1.5 → -1.5	799	0.000	yes
horizontal-09	944–975	reverse	−1.5 → +1.5	948	1.521	no — exception
horizontal-10	1039–1079	reverse	+1.5 → -1.5	1048	0.000	yes
horizontal-11	1131–1156	stop	−1.5 → +0	1139	0.000	yes
horizontal-12	1195–1220	start	+0 → +1.5	1202	0.000	yes
horizontal-13	1318–1346	stop	+1.5 → +0	1326	0.000	yes
horizontal-14	1349–1377	start	+0 → -1.5	1359	0.000	yes
horizontal-15	1485–1512	stop	−1.5 → +0	1494	0.000	yes
horizontal-16	1507–1534	start	+0 → +1.5	1515	0.000	yes
horizontal-17	1602–1630	stop	+1.5 → +0	1610	0.000	yes
horizontal-18	1622–1650	start	+0 → -1.5	1632	0.000	yes
horizontal-19	1716–1756	reverse	−1.5 → +1.5	1725	0.000	yes
horizontal-20	1833–1852	stop	+1.5 → +0	1838	0.632	no — exception

Exceptions are not used to redefine the shared acceleration:

- **H06 / H09:** faster-than-model stopping or turning around the same leftmost world-box X≈185. This is compatible with a positional constraint/collision or additional input behavior. No key log or decoded collision rule identifies the cause. A constant-floor experiment does not guarantee the absence of horizontal constraints. Retain these traces as explicit counterexamples to a universal free-motion-only model.
- **H20:** the character becomes stationary while still above the floor, before the tutorial/dialogue overlay appears. Both axes freeze. This is not a clean free-motion braking measurement; the final partial jump is excluded from the 15 complete jumps. The precise triggering event remains unknown.

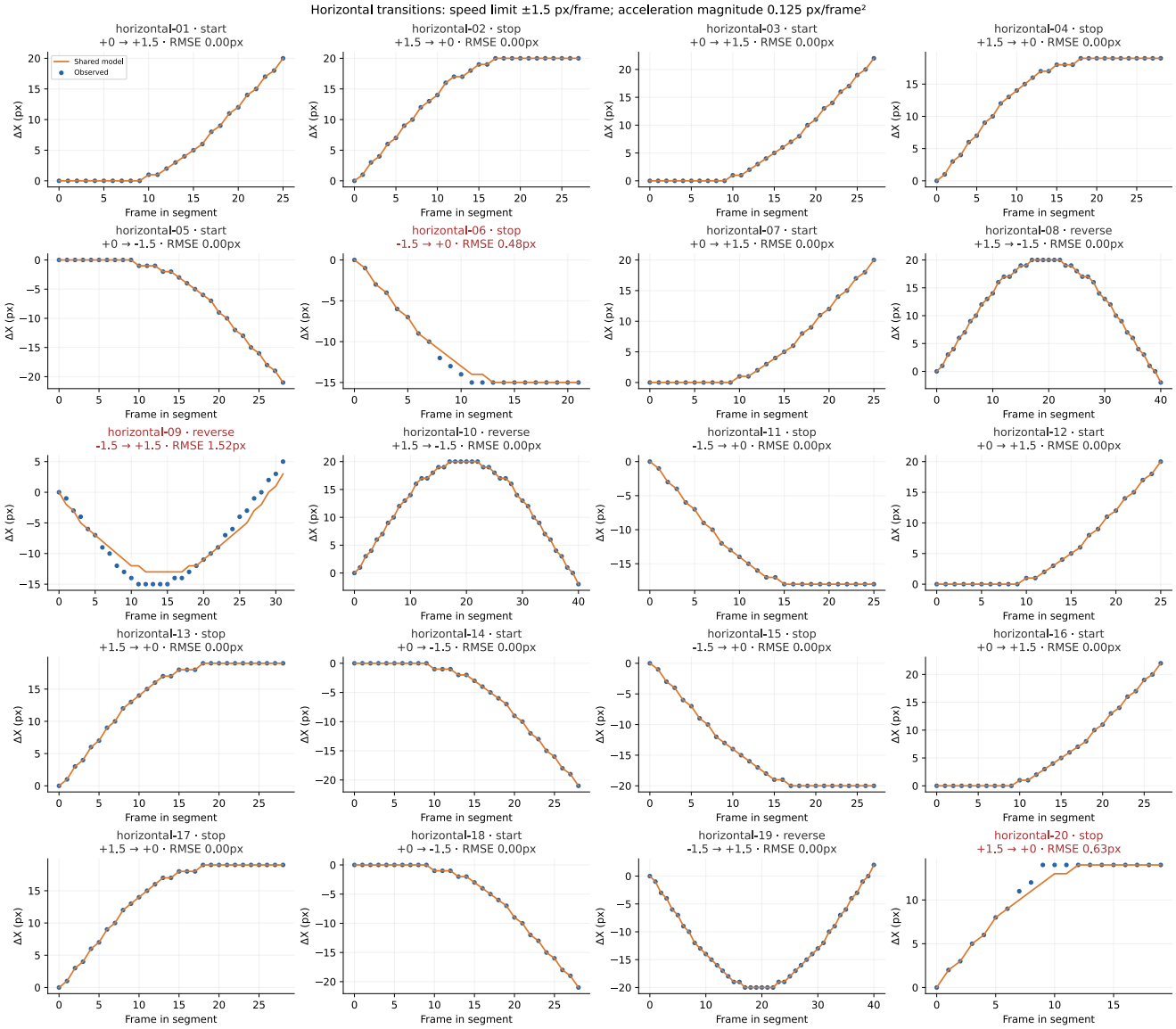

Figure 3: Horizontal position fits

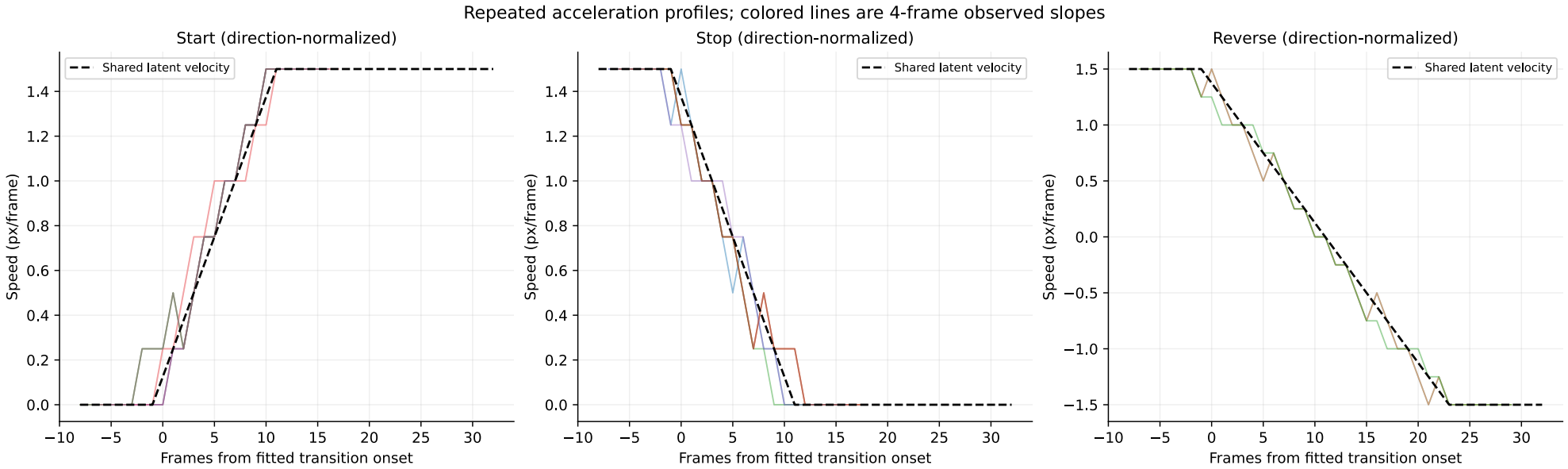

Figure 4: Repeated acceleration

Repeatability and prediction checks

No per-jump gravity, launch speed or release-speed cap was allowed. No per-event horizontal acceleration or terminal running speed was allowed. The only fitted nuisance variables were the event's integer switch frame and subpixel position phase. The match is therefore more restrictive than fitting an independent polynomial to every occurrence.

Independent quadratic fits are retained in `measurements.json` as a comparison. They approximate the full jumps, but leave structured errors in the shortened ascent and at terminal-speed descent. The shared piecewise discrete model explains both features and has zero pixel residual on all complete jump traces.

As an additional test, each jump's phase and early-release event were calibrated using **only its first eight airborne samples**, then the rest was predicted, assuming no additional later release that would change the path. This does not refit physics constants and does not retune phase on the later samples.

Of **716 later samples (including landing)**, **668 (93.3%)** are exact using the earliest compatible phase; **all are within one pixel**. Remaining differences reflect unresolved subpixel phase in a short integer-pixel prefix. This is a conditional suffix-prediction check, not a blind new recording: the shared model was developed on this clip.

Perturbing the shared gravity while refitting phase/release gives:

Gravity (px/frame²)	Aggregate airborne position RMSE (px)
0.1093750	2.0511
0.1171875	1.7658
0.1250000	0.0000
0.1328125	5.9130
0.1406250	10.8357

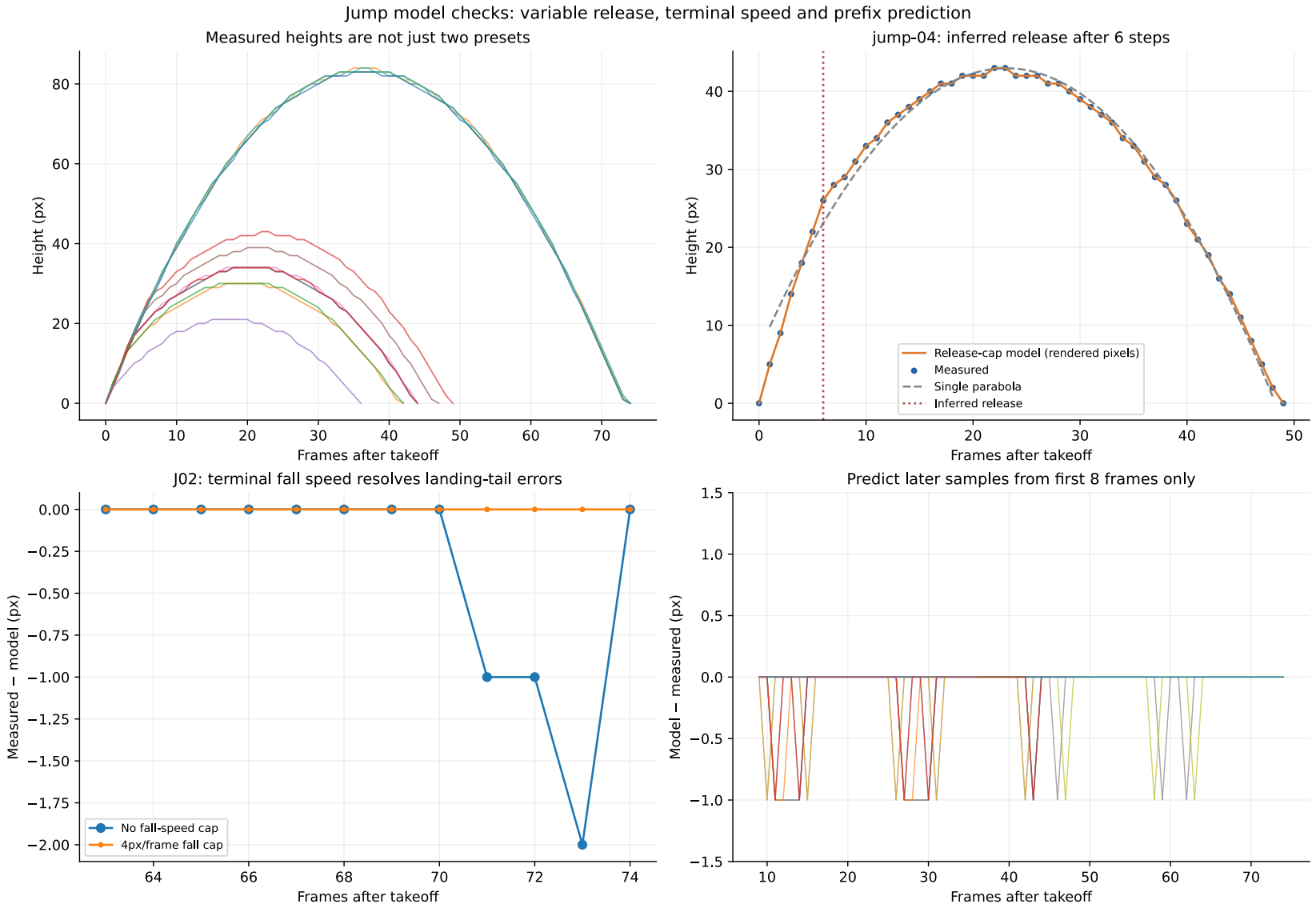

Figure 5: Model checks and prefix predictions

Candidate simulation ordering

This pseudocode expresses a model compatible with the free-motion samples, not the recovered original input/collision implementation. One step corresponds to one recorded frame interval in this experiment.

```
vx = approach(vx, 1.5 * direction, 0.125)
x += vx

if jump_started_on_ground:
    vy = -4.5
if jump_released_or_not_held and vy < -2:
    vy = -2
y += vy
vy = min(vy + 0.125, 4)

resolve_ground_and_other_collisions()  # not identified by this fitting experiment
render_full_sprite_box(floor(x), floor(y))
```

Ordering of checks around the ground collision, jump retriggering/buffering, input sampling, and phase persistence through landing remains unverified. Equivalent models can shift initialization by a tick. Keep these constants as a candidate profile until tested on a new, controlled recording.

Synchronized input recording and next experiment

For frame-accurate input attribution, instrument the emulator to write a sidecar log at the keyboard-event delivery point and the video-capture frame submission point, using the **same emulated clock**. Log capture-session ID, capture-frame index, emulated timestamp, key down/up events, and held-key state. Preserve the exact one-to-one relation with AVI frames, including duplicate/empty frames and capture restarts. If possible, also trace when the game reads input, because an event delivered to the emulated keyboard is not necessarily consumed immediately.

This is a proposed instrumentation change, not a feature added in this task. DOSBox-X exposes the emulated timing primitive `PIC_FullIndex`; its code separates host and emulated time. A desktop-only key logger can help, but needs clock/offset calibration and cannot establish exact game-consumption timing on its own. Reference: DOSBox-X timing source.

Record a controlled sweep away from the leftmost X≈185 and tutorial trigger:

1. Jump from rest with scheduled releases after 1, 2, 3, 4, 5, 6, 8, 12, 16, 20, 24 and 36 emulated frames; repeat each at least three times.
2. From rest, hold right until full speed, release, and wait until stationary; repeat left. Then reverse directly from full speed in both directions.
3. Repeat starts/stops/reversals during ascent and descent.
4. Freeze the current model parameters before scoring that new recording. Compare whole trajectories, event-to-response delay and residuals, not only peak heights.

The current measurements strongly support the six shared constants and the release-speed-cap mechanism. The next recording should test input timing and the three horizontal exceptions, rather than merely refitting the same data.

Files and reproducibility

- `index.html`: plot gallery, written report and individually playable clips.
- `plots/*.png` and `plots/*.svg`: standalone figures.
- `segments/*.csv`: per-occurrence source frame/PTS, timestamps and world boxes.
- `clips/*.mp4`: fixed world-space crops stabilized with the measured camera; original frame cadence retained, full sprite box overlaid, 2× nearest-neighbor. Black margins mean the world region was outside the recorded viewport. These are compressed previews; no fitting uses their pixels.
- `measurements.json`: all segment bounds, fit errors, phases, inferred event frames, independent polynomial coefficients and validation statistics.
- `model.json`: shared candidate constants, separate from the remake engine.

The complete video track remains in `../movement/`. Regenerate this study from the repository root with Python, NumPy, Pillow, Matplotlib and FFmpeg available:

```
python tools/video-analyzer/study.py outputs/video-analysis/movement --model tools/video-analyzer/profiles/
kellogg-motion-candidate.json --output outputs/video-analysis/movement-study-new
```