COMMITSTREAM: CAN MODELS KEEP THEIR ANSWERS CORRECT IN STREAMING VIDEO?

Anonymous authors
Paper under double-blind review

ABSTRACT

Advances in embodied intelligence and world modeling place increasing demands on models' video understanding. For video assistants, a key challenge is that an answer that is correct when issued can become wrong as the scene changes. If an assistant neither updates nor withdraws it, users may continue acting on outdated information. Reliable assistance therefore requires maintaining an answer that is supported by the available evidence throughout an ongoing task. We introduce CommitStream, a benchmark that jointly evaluates when models should wait for evidence, provide an answer, and revise it as events unfold. To support this evaluation, a reusable pipeline generates and refines annotations of when answers become supported and when they must change, followed by a final human review. The resulting benchmark contains 1,006 tasks from 307 video episodes across six domains. Evaluation follows causal real-time playback, using streaming inference for models with a streaming input interface and asynchronous sliding-window inference for those without one. Under the official ± 1 s tolerance, the best model with streaming inference remains correct throughout 19.88% of tasks; the best with asynchronous sliding-window inference reaches 29.03%. Even the latter leaves stale or wrong answers in effect for 13.94% of scored time. These results identify sustained answer reliability as an unresolved challenge and establish CommitStream as a testbed for studying when video assistants should answer, wait, and update. Code and data are available at `https://anonymous.4open.science/r/commitstream`.

1 INTRODUCTION

Live video understanding requires an assistant to maintain information that users can rely on as events unfold. An answer that is correct when issued can become outdated as the scene changes. Until the assistant updates or withdraws it, users may continue acting on information that is no longer supported. Reliable assistance therefore requires deciding when available evidence is sufficient to answer, keeping that answer current, and recognizing when it should be withdrawn. These decisions connect the correctness of an individual response to the reliability of assistance throughout an ongoing task.

Some benchmarks assess responses only at selected timestamps (Xun et al., 2026), while others focus on restricted forms of state maintenance, such as counting (Liu et al., 2026). Assessing sustained answer reliability also requires tracking whether the answer available to the user remains supported as the video unfolds, including between model responses.

We introduce CommitStream, a benchmark that jointly evaluates when models should wait for evidence, provide an answer, and revise it as events unfold. A task issued during playback asks the model to maintain one fact as new frames arrive. The answer currently available to the user is the model's *public state*: silence preserves it, while explicit defer withdraws it. Reference annotations specify when each answer becomes supported and when it must change, allowing the model's behavior to be assessed throughout the task (Figure 1).

We construct these reference trajectories with a staged annotation pipeline that separates task-definition validation from temporal annotation and concludes with human review (Figure 2). The

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

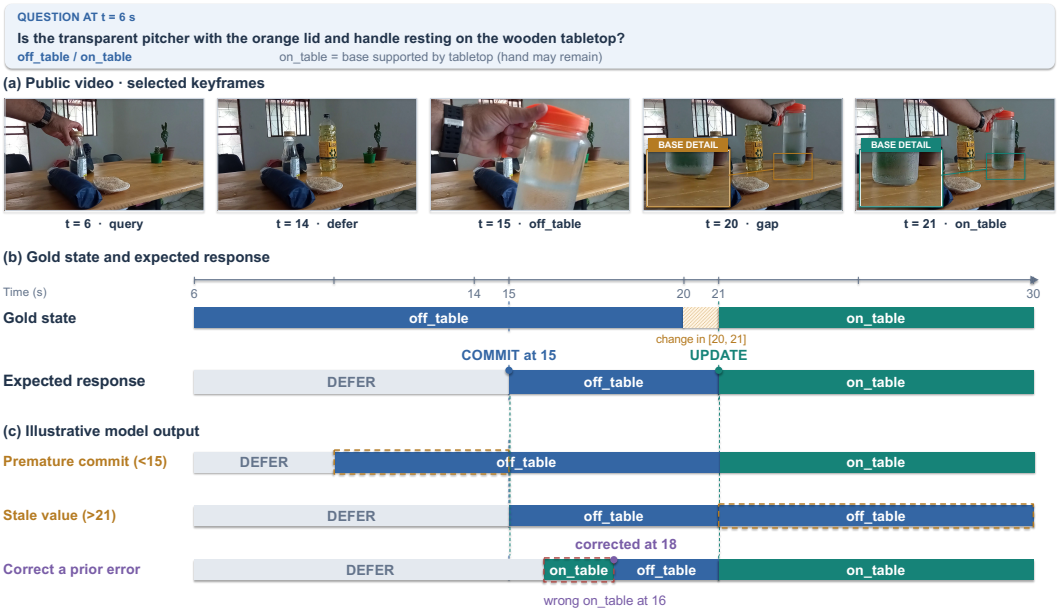


Figure 1: **CommitStream evaluates when to wait, answer, and update.** (a) A task issued at 6 s asks whether the pitcher rests on the table. (b) The model should defer until sufficient evidence becomes available at 15 s, commit to `off_table`, and update to `on_table` at 21 s. The reference trajectory distinguishes when a state holds from when it is sufficiently supported to answer. (c) Illustrative model outputs show premature commitment, a stale answer, and correction of a prior error. Silence preserves the public state, so an incorrect answer remains in effect until updated or withdrawn.

resulting benchmark contains 1,006 tasks from 307 video episodes across six domains and five temporal modes: current, cumulative, stable, last-completed, and first-event.

Evaluation follows causal real-time playback, with responses taking effect when generation finishes. Models with a streaming input interface use streaming inference, while models without one are evaluated through asynchronous sliding-window inference that repeatedly submits recent frames together with the current public state. Across both protocols, current models rarely maintain a fully correct trajectory and exhibit distinct failures in waiting, commitment, and updating. Performance also varies substantially across temporal modes, indicating that maintaining accumulated or changing information remains especially challenging.

Our contributions are:

- **A task for sustained answer reliability.** Models maintain an answer throughout an ongoing video task, with reference annotations supporting evaluation of when to wait, commit, and update.
- **A reusable annotation pipeline.** Task validation, timeline construction, refinement, and review achieve 97.96% task-level annotation accuracy against human annotations on test tasks at ± 1 tick. Final human review completes benchmark construction.
- **A benchmark spanning five temporal modes.** The dataset contains 1,006 tasks from 307 episodes across six domains, with task definitions, supporting visual evidence, and reference trajectories for maintaining answers over time.
- **Evaluation across model interfaces.** We evaluate nine models with streaming inference and adapt five models without a streaming input interface through asynchronous sliding-window inference, revealing failures in waiting, commitment, and updating.

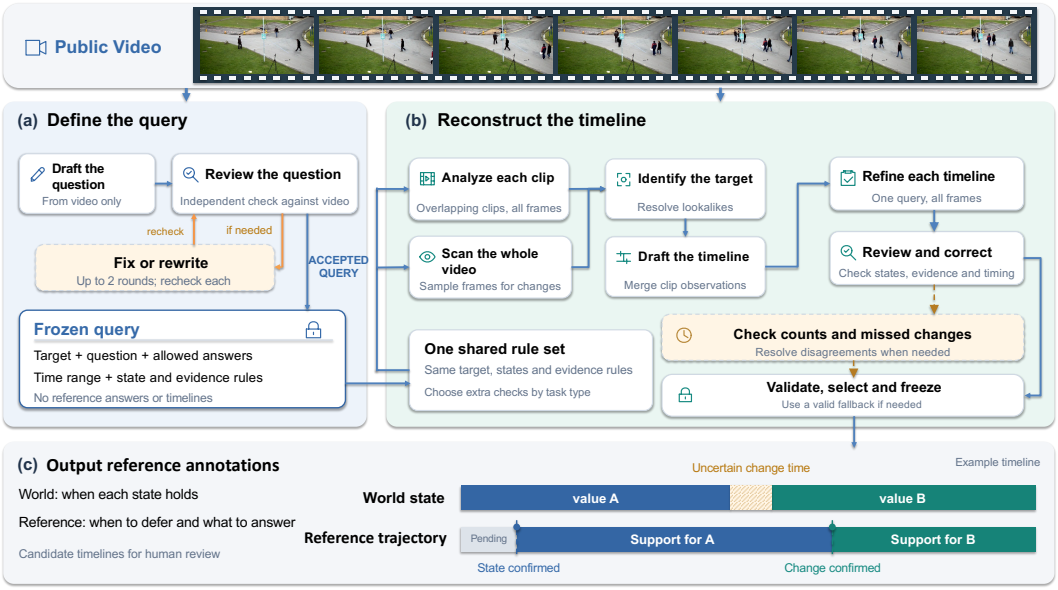


Figure 2: **Temporal annotation pipeline.** (a) Task definitions are reviewed, revised, and frozen. (b) Local and global observations guide timeline drafting, refinement, and review. Validation selects a valid candidate, with fallback if needed; dashed arrows indicate conditional checks. (c) Annotations record both when states hold and when they are supported, yielding the reference trajectory used for evaluation. Two authors review the reference annotations against the videos; tasks with incorrect annotations are discarded.

2 The CommitStream Task

A task specifies a target, a question, allowed labels $\mathcal{V}_i$, label definitions, and rules for scope, visibility, and deciding between labels. Within a video window $[s_i, e_i)$, the question is issued at q_i and remains active until e_i. Evaluation covers the integer seconds $T_i = \{q_i, \ldots, e_i - 1\}$.

The *public state* $p_i(t) \in \mathcal{V}_i \cup \{\perp\}$ is the answer currently available to the user, initially defer ($\perp$). An allowed label establishes or updates it; `__DEFER__` withdraws it; `<|silence|>` preserves it. Silence can therefore retain either defer or a previously issued answer. Responses contain only an allowed label, `__DEFER__`, or `<|silence|>`; explanations and end-of-video summaries do not satisfy the protocol.

Reference trajectory. Annotations record both when a state holds and when the available frames provide enough evidence to answer. In Figure 1, for example, the model should defer before it can identify the target pitcher. Evidence follows the task's rules: a completed event can remain established after its participants leave the image.

These annotations yield the *reference trajectory* $g_i(t) \in \mathcal{V}_i \cup \{\perp\}$ used for scoring. It is defer before an answer is supported and gives the supported label otherwise. We refer to its intervals by the action they require: defer, commit to a supported label, or update a previous label. These intervals cover the entire scored window without gaps or overlaps, and adjacent intervals with the same state are merged.

3 Benchmark Construction

3.1 Data Collection and Screening

The benchmark draws on nine video sources: Wikimedia Commons, CaptainCook4D, Perception Test, DSEC, MEVA, Breakfast Actions, the Leeds Complex and Long Activities Dataset, NOAA Fisheries, and DVIDS (Peddi et al., 2024; Patraucean et al., 2023; Gehrig et al., 2021; Corona et al.,

Table 1: **Temporal modes and their task counts in CommitStream.**

Temporal mode	Tasks	Information to maintain
current	467	The target's current state or relation.
cumulative	252	Events accumulated since the query, including counts and whether an event has occurred.
stable	183	A fixed target property once sufficient evidence identifies it.
last-completed	79	The state or attribute of the most recently completed event.
first-event	25	The outcome or attribute of the first specified event after the query.

2021; Kuehne et al., 2014; Tayyub et al., 2017). Videos are screened for traceable provenance, successful decoding, target visibility, and suitability for the task. Source and scene records are used to identify duplicate candidates. Each retained episode is extracted as a continuous video window and sampled at one-second intervals, with source-specific attribution and timing records preserved.

3.2 Annotation Construction and Human Review

The pipeline (Figure 2) first drafts task definitions and checks them against the video. Candidates can undergo up to two repair or rewriting rounds, each followed by rechecking; unresolved candidates are withheld. Accepted task definitions are frozen before temporal annotation.

The episode is analyzed in 32-frame clips overlapping by four frames, together with a global scan that samples every fourth second plus the query and endpoints. These observations help distinguish similar objects and draft candidate annotations of the state and the evidence available for it.

Two further passes inspect all sampled frames per task: the first checks content and temporal coverage, and the second checks states, evidence, and timing together. Conditional checks address counts, missed changes, and disagreements. A final validation step checks structure and consistency with the task definition, using a valid upstream result as fallback if needed. Automated stages use GPT-6-Astra with high reasoning effort in separate contexts. The validated annotations are converted into the reference trajectory used for evaluation.

Human review. Two authors review the candidate tasks and their reference annotations against the corresponding videos and task definitions. They assess whether the answer labels are correct and whether the reference trajectory specifies when to defer, commit, and update in accordance with the available visual evidence. Tasks with incorrect reference annotations are discarded, and only tasks judged correct are retained in the benchmark. Each retained task includes its definitions, timestamped visual evidence, and reference trajectory.

3.3 Dataset Coverage

The final benchmark contains **1,006 tasks from 307 episodes**. Its six domains comprise operations (452 tasks), animals (157), traffic (111), mixed scenes (108), people (100), and sports (78). Table 1 defines the information maintained in each temporal mode.

The reference trajectories contain 142 defer, 1,011 commit, and 1,125 update segments, totaling 57,277 scored seconds. There are 693 tasks with an update segment and 141 with a defer segment. Five tasks return to defer after a commitment.

4 Evaluation Protocol and Metrics

4.1 Causal Real-Time Playback

All runs use chronological frames at 1 FPS and $1\times$ playback: each video second takes one wall-clock second. The task is introduced with the query frame. Before the query, models using streaming inference receive frames with the fixed instruction, "Observe the incoming live video silently. A maintenance task will be issued later." Inference uses only videos and task definitions; reference annotations are available only to the evaluator.

For streaming inference, arrived frames are processed through streaming or causal recomputation interfaces. Playback continues during generation. A response takes effect at completion, mapped to video time and rounded down to an integer second; responses outside the scored window have no effect.

Asynchronous sliding-window inference. Models without a streaming input interface accept a fixed set of frames per request. This protocol adapts them to ongoing video through successive requests while playback continues. Frames are buffered before the query without issuing requests. From the query onward, a request is scheduled every 4 s, containing the task, the latest at most 16 arrived frames in chronological order, and the model's latest applied public state. The frame indices identify their video times. State feedback carries the model's own current answer into the next request as the visual window moves.

At each scheduled slot, a request is submitted regardless of earlier requests still in flight; slots missed because of scheduler overruns are logged. Responses take effect at completion. An older request cannot overwrite a public state already updated by a newer request. Silence preserves the public state, and invalid responses do not replace it. Frames, requests, completions, and failures are logged, with one prediction record per task. We use the implementation detailed in Appendix B.

4.2 From Per-Second Correctness to Task Reliability

Timing tolerance and per-second correctness. At each scored second, the evaluator compares the model's public state with the reference trajectory. The official timing tolerance is ± 1 s: immediately around a reference transition, either adjacent reference state is accepted. For a transition at 21 s, for example, both states are accepted at seconds 20 and 21. Other seconds require the reference state, subject to any other adjacent transition. Every scored second remains in the denominator. An invalid response makes its event second incorrect while leaving the persistent public state unchanged.

For task i, let $c_i(t)$ be 1 if the state at scored second t is accepted under this rule, and 0 otherwise. All metrics below use this same correctness decision. We additionally rescore each run at ± 2 s and ± 3 s to measure sensitivity to timing tolerance.

D-Score, C-Score, and U-Score. These scores assess waiting, commitment, and updating, respectively. First, each reference segment receives the percentage of its seconds that are correct. For a segment S belonging to task i, with $|S|$ scored seconds, this percentage is

$$a(S) = \frac{100}{|S|} \sum_{t \in S} c_i(t). \tag{1}$$

Next, segment percentages are averaged within each role across the dataset. Let $\mathcal{S}_r$ be the set of all reference segments whose role is r, where r is defer, commit, or update. Then

$$\mathrm{Score}_r = \frac{1}{|\mathcal{S}_r|} \sum_{S \in \mathcal{S}_r} a(S). \tag{2}$$

The three resulting values are D-Score, C-Score, and U-Score. A segment receives equal weight to other segments of the same role, regardless of its length. Their arithmetic mean, reported as the *Mean Role Score*, gives the three roles equal weight.

Exact Trajectory Rate. *Exact Trajectory Rate* (ETR) measures whether an answer remains correct throughout a task. A task succeeds only if $c_i(t) = 1$ at every scored second in its window T_i. ETR is the percentage of evaluated tasks that succeed. One incorrect second makes a task unsuccessful. For analysis by temporal mode, we apply this same definition separately to current, cumulative, stable, last-completed, and first-event tasks. These are task types, distinct from the three segment roles used by D-Score, C-Score, and U-Score.

Error exposure. To measure how long incorrect answers remain available, we classify every scored second descriptively. A second is correct when its state is accepted; otherwise, it is invalid for a malformed response, premature for answering during defer, overdue for remaining deferred after an answer is supported, stale for retaining the previous supported label, or wrong for another

Table 2: **Official ± 1 s results on all 1,006 tasks.** All entries are percentages. Higher is better except for stale + wrong, which measures incorrect-answer exposure. Bold marks the highest ETR, Mean Role Score, and role scores within each inference protocol.

System	ETR	Mean Role Score	D-Score	C-Score	U-Score	Stale + wrong ↓
Streaming inference						
MOSS-VL-Realtime	14.02	**57.91**	68.90	47.61	**57.22**	34.82
TimeChatOnline-7B	3.98	48.74	56.74	47.67	41.82	36.38
VST-7B	**19.88**	44.13	11.01	**75.61**	45.76	37.40
MiniCPM-o-4.5	8.55	43.99	38.74	48.58	44.65	32.15
ReKV	6.26	38.08	69.66	31.09	13.49	20.46
LiveCC-7B-Instruct	13.12	28.33	7.82	44.40	32.79	34.44
MMDuet2	0.70	26.00	59.67	10.29	8.05	7.92
Flash-VStream-Qwen-7b	10.93	25.57	7.13	34.98	34.60	42.21
VideoLLM-online-8B-v1plus	0.00	24.32	**70.92**	2.00	0.04	0.00
Asynchronous sliding-window inference						
Qwen3.8-Flash-Next	**29.03**	**72.54**	77.17	70.89	69.56	13.94
GLM-5.3-Flash	8.35	71.87	84.46	60.43	**70.72**	13.53
dots3-note-prev	24.95	67.69	62.20	**75.25**	65.62	19.70
MiMo-V2.6-Flash-RL	14.71	58.59	81.38	51.62	42.78	27.36
InternVideo3-8B-Instruct	9.05	52.28	**85.68**	36.60	34.55	18.33

incorrect label. Error exposure is the percentage of scored seconds in each case. We also report stale + wrong, the combined percentage during which an outdated or incorrect label remains in effect after an answer is supported.

Correction outcomes. Within each commit or update interval, we track each period during which the model maintains an incorrect allowed label. The period ends when the model either repairs the answer by switching to an accepted label or safely retracts it by returning to defer; otherwise it remains unrepaired when the reference interval ends. Outcome percentages use all such periods as their denominator.

5 Experiments

5.1 Experimental Setup

We report nine models using streaming inference (Wang et al., 2026a; Yao et al., 2025; Guan et al., 2026; Cui et al., 2026; Di et al., 2025; Chen et al., 2025b; Wang et al., 2026c; Zhang et al., 2025; Chen et al., 2024) and five locally deployed models using asynchronous sliding-window inference (Table 2) (Qiu et al., 2026; Yan et al., 2026). Every run covers the same 1,006 tasks with no duplicate or failed records. Checkpoints, processors, prompts, and adapters are fixed within each run. Models using streaming inference use greedy decoding and their recorded history or memory interfaces. Asynchronous sliding-window inference uses the schedule, with 16-frame inputs and at most 262,144 pixels per frame. Appendix B specifies request construction and completion handling. Appendix D presents recorded model trajectories on a shared example.

We exclude proprietary models accessible only through APIs because frequent rate limiting makes sustained real-time evaluation unreliable, while opaque and potentially dynamic backend routing prevents the resulting performance from being attributed to a fixed model version or configuration.

5.2 Results: Waiting, Commitment, and Sustained Correctness

Table 2 shows that the highest ETR is 19.88% with streaming inference (VST-7B) and 29.03% with asynchronous sliding-window inference (Qwen3.8-Flash-Next). Most tasks therefore contain

Table 3: **Sensitivity to timing tolerance.** All entries are percentages on the same 1,006 tasks. Each row rescores fixed predictions; the ±1 s columns are official.

System	Mean Role Score			ETR		
	±1 s	±2 s	±3 s	±1 s	±2 s	±3 s
Streaming inference						
MOSS-VL-Realtime	57.91	61.40	64.15	14.02	15.71	17.99
TimeChatOnline-7B	48.74	52.07	54.54	3.98	4.37	5.17
VST-7B	44.13	49.16	52.76	19.88	23.76	25.55
MiniCPM-o-4.5	43.99	47.12	49.42	8.55	10.04	10.74
ReKV	38.08	39.99	41.34	6.26	7.26	8.05
LiveCC-7B-Instruct	28.33	31.73	34.51	13.12	15.01	16.00
MMDuet2	26.00	26.91	27.65	0.70	0.70	0.70
Flash-VStream-Qwen-7b	25.57	28.87	31.37	10.93	12.23	13.22
VideoLLM-online-8B-v1plus	24.32	24.69	25.02	0.00	0.00	0.00
Asynchronous sliding-window inference						
Qwen3.8-Flash-Next	72.54	75.46	77.31	29.03	35.88	42.05
GLM-5.3-Flash	71.87	75.19	77.24	8.35	10.64	13.32
dots3-note-prev	67.69	71.50	74.04	24.95	30.22	34.39
MiMo-V2.6-Flash-RL	58.59	61.60	63.83	14.71	16.60	17.79
InternVideo3-8B-Instruct	52.28	55.12	57.11	9.05	10.34	11.53

at least one incorrect scored second even for the best model in each group. The role scores reveal different difficulties: VST-7B reaches C-Score 75.61% but D-Score 11.01%, whereas MOSS-VL-Realtime reaches D-Score 68.90% but C-Score 47.61%. Successfully establishing an answer and waiting until evidence supports it remain distinct challenges.

Qwen3.8-Flash-Next reaches a Mean Role Score of 72.54%, with all three role scores above 69%. GLM-5.3-Flash has a similar Mean Role Score of 71.87% and a higher U-Score, 70.72% versus 69.56%, but its ETR is 8.35%. Good average correctness within reference intervals can coexist with errors that interrupt many complete trajectories.

5.3 Sensitivity to Timing Tolerance

We rescore the same predictions at ±1, ±2, and ±3 s (Table 3). Increasing tolerance from ±1 to ±3 s raises Qwen3.8-Flash-Next's ETR from 29.03% to 42.05% and VST-7B's from 19.88% to 25.55%. Some complete-task failures are therefore confined to the additional seconds accepted around reference transitions, while most tasks remain unsuccessful even at ±3 s.

5.4 Error Exposure and Correction Outcomes

Table 2 and Figure 3(a) quantify how incomplete trajectories affect the answer available to the user. Stale or wrong answers occupy 34.82% of scored time for MOSS-VL-Realtime and 37.40% for VST-7B. Qwen3.8-Flash-Next has lower exposure, 13.94%, but still maintains incorrect answers during part of the task. InternVideo3-8B-Instruct spends 18.33% of scored time stale or wrong and 38.67% overdue. Its errors thus also include substantial time without an answer after supporting evidence is available.

Figure 3(b) examines whether models revise an incorrect public state. Qwen3.8-Flash-Next repairs 39.02% of its incorrect-answer periods and retracts to defer in 1.48%; 59.50% remain unrepaired before the corresponding reference interval ends. GLM-5.3-Flash has the highest repair rate among the reported systems, 55.02%, while its ETR remains 8.35%. The two measures describe complementary behavior: repairing an error after it occurs and maintaining a correct answer throughout the task.

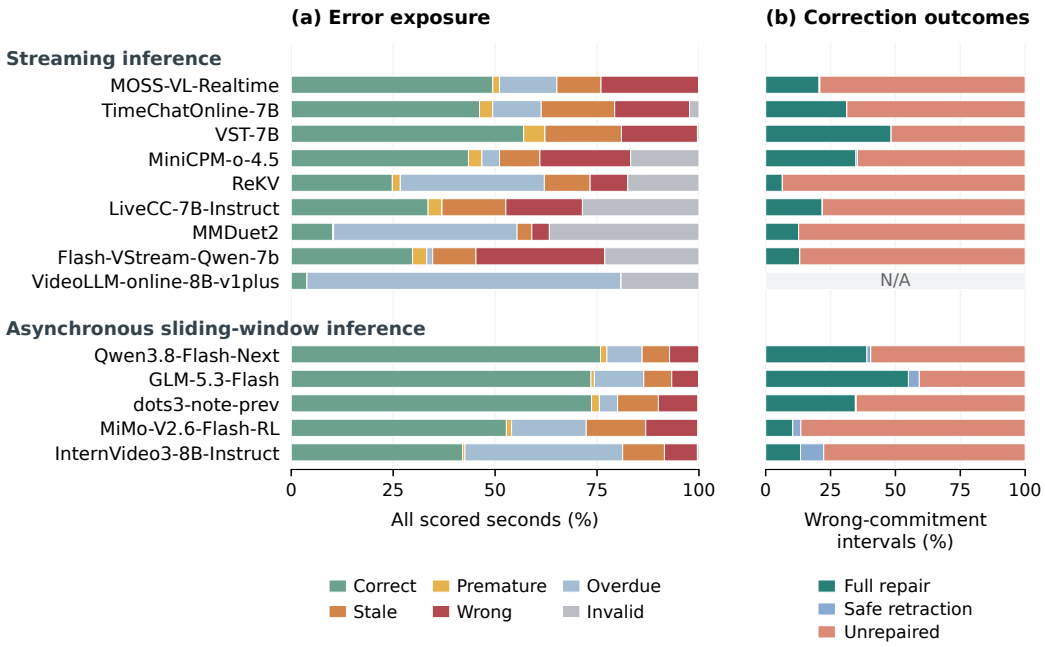


Figure 3: **Error exposure and correction outcomes at ± 1 s.** (a) Percentages of scored seconds that are correct, premature, overdue, stale, wrong, or invalid. (b) Outcomes of periods in which a model maintains an incorrect answer: repaired, retracted to defer, or unrepaired. Each panel groups the 14 models by inference protocol. N/A denotes that no such periods occur.

Table 4: **ETR by temporal mode at ± 1 s.** Each percentage is the fraction of tasks in that mode that remain correct at every scored second.

System	current	cumulative	stable	last-completed	first-event
Streaming inference					
MOSS-VL-Realtime	11.35	7.14	34.43	1.27	24.00
TimeChatOnline-7B	4.28	1.98	7.10	0.00	8.00
VST-7B	14.99	6.75	60.11	2.53	4.00
MiniCPM-o-4.5	4.28	3.97	29.51	0.00	8.00
ReKV	2.57	0.79	25.68	0.00	8.00
LiveCC-7B-Instruct	10.28	3.17	40.44	1.27	4.00
MMDuet2	1.07	0.00	1.09	0.00	0.00
Flash-VStream-Qwen-7b	9.64	1.59	31.69	2.53	4.00
VideoLLM-online-8B-v1plus	0.00	0.00	0.00	0.00	0.00
Asynchronous sliding-window inference					
Qwen3.8-Flash-Next	25.70	18.25	64.48	5.06	16.00
GLM-5.3-Flash	5.78	1.98	25.14	0.00	24.00
dots3-note-prev	21.20	9.52	66.67	5.06	8.00
MiMo-V2.6-Flash-RL	13.28	1.19	43.72	1.27	8.00
InternVideo3-8B-Instruct	7.07	1.98	26.78	1.27	12.00

5.5 Sustained Correctness Across Temporal Modes

We report ETR separately for the five temporal modes to examine how sustained answer reliability varies across task semantics. As Table 4 shows, Qwen3.8-Flash-Next reaches 64.48% on stable

Table 5: **Annotation pipeline ablation.** Task-level annotation accuracy (%) against human reference annotations. The third row jointly removes refinement and review; the last disables only the fallback branch.

Configuration	±1 tick	±2 ticks	±3 ticks
Full	**97.96**	**97.96**	**97.96**
Full − Review and correct	73.47	79.59	83.67
Full − Refine each timeline − Review and correct	71.43	77.55	79.59
Full − valid fallback	87.76	87.76	87.76

tasks but 18.25% on cumulative tasks and 5.06% on last-completed tasks. VST-7B reaches 60.11% on stable tasks and 6.75% on cumulative tasks. All five models using asynchronous sliding-window inference have lower ETR on cumulative and last-completed tasks than on stable tasks. Maintaining an accumulated result or the latest completed event remains difficult in these groups; the comparison is descriptive because their content and update patterns also differ.

6 Annotation Pipeline Ablation

We compare four automated pipeline configurations on the same tasks. Task-level annotation accuracy is the percentage agreeing completely with human reference annotations in answer content and timing under the stated tolerance. One tick is one sampled video second. The full configuration includes every automated stage in Figure 2. One ablation removes the final review pass, another removes both refinement and review, and the last disables the fallback to a valid upstream annotation while retaining validation. The full configuration reaches 97.96% accuracy at every tolerance (Table 5). Removing review lowers ±1 tick accuracy by 24.49 percentage points; wider tolerance recovers part of this loss. Disabling fallback lowers accuracy to 87.76%, showing the value of retaining valid upstream annotations when later outputs fail validation.

7 Related Work

LongVideoBench and MMBench-Video evaluate video question answering, while ViLMA and CG-Bench examine temporal grounding and supporting evidence (Wu et al., 2024; Fang et al., 2024; Kesen et al., 2024; Chen et al., 2025a). SVBench and RIVER extend evaluation to temporally connected dialogue and real-time interaction (Yang et al., 2025; Shi et al., 2026). RTV-Bench scores responses at selected timestamps, leaving the answer maintained between queries unscored (Xun et al., 2026).

SVCBench studies state maintenance through counting, while MedStreamBench studies response timing and post-evidence stability in medical video (Liu et al., 2026; Wang et al., 2026b). CommitStream adds explicit withdrawal through defer, evaluates defer, commit, and update segments separately across five temporal modes, and reports complete-task correctness and error exposure.

Selective prediction studies the trade-off between answering more inputs and reducing errors through abstention (Geifman & El-Yaniv, 2019; Whitehead et al., 2022). In CommitStream, abstention is a time-dependent public state: a model must also recognize when evidence becomes sufficient and when a prior commitment must change or be withdrawn.

8 Conclusion

CommitStream evaluates the reliability of answers maintained during live video understanding. It brings waiting, commitment, and updating into a unified task formulation. Across 1,006 tasks from 307 video episodes, experiments reveal complementary challenges in waiting for sufficient evidence, keeping answers current, and revising incorrect public states. By examining both complete-task correctness and how long incorrect answers remain in effect, CommitStream makes these challenges measurable and provides a testbed for improving sustained answer reliability in video assistants.

AI USE STATEMENT

We used generative AI tools as components of the benchmark annotation pipeline, including task-definition validation, temporal annotation, refinement, and consistency checking, as described in Section 3. All retained tasks and reference trajectories underwent final human review, and tasks with incorrect annotations were discarded. Generative AI tools were also used to assist with code development and language editing. The authors reviewed and verified all AI-assisted outputs and take responsibility for the final content of this work, including its text, claims, code, and benchmark artifacts.

REFERENCES

Guo Chen, Yicheng Liu, Yifei Huang, Baoqi Pei, Jilan Xu, Yuping He, Tong Lu, Yali Wang, and Limin Wang. Cg-bench: Clue-grounded question answering benchmark for long video understanding. In *International Conference on Learning Representations*, volume 2025, pp. 45647–45682, 2025a.

Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. Videollm-online: Online video large language model for streaming video. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18407–18418. IEEE, 2024.

Joya Chen, Ziyun Zeng, Yiqi Lin, Wei Li, Zejun Ma, and Mike Zheng Shou. Live: Learning video llm with streaming speech transcription at scale. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29083–29095. IEEE, 2025b.

Kellie Corona, Katie Osterdahl, Roderic Collins, and Anthony Hoogs. Meva: A large-scale multiview, multimodal video dataset for activity detection. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1059–1067. IEEE, 2021.

Junbo Cui, Bokai Xu, Chongyi Wang, Tianyu Yu, Weiyue Sun, Yingjing Xu, Tianran Wang, Zhihui He, Wenshuo Ma, Tianchi Cai, et al. Minicpm-o 4.5: Towards real-time full-duplex omni-modal interaction. *arXiv preprint arXiv:2604.27393*, 2026.

Shangzhe Di, Zhelun Yu, Guanghao Zhang, Haoyuan Li, Hao Cheng, Bolin Li, Wanggui He, Fangxun Shu, and Hao Jiang. Streaming video question-answering with in-context video kv-cache retrieval. In *International Conference on Learning Representations*, volume 2025, pp. 42115–42127, 2025.

Xinyu Fang, Kangrui Mao, Haodong Duan, Xiangyu Zhao, Yining Li, Dahua Lin, and Kai Chen. Mmbench-video: A long-form multi-shot benchmark for holistic video understanding. *Advances in Neural Information Processing Systems*, 37:89098–89124, 2024.

Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. Dsec: A stereo event camera dataset for driving scenarios. *IEEE Robotics and Automation Letters*, 6(3):4947–4954, 2021.

Yonatan Geifman and Ran El-Yaniv. Selectivenet: A deep neural network with an integrated reject option. In *International conference on machine learning*, pp. 2151–2159. PMLR, 2019.

Yiran Guan, Liang Yin, Dingkang Liang, Jianzhong Ju, Zhenbo Luo, Jian Luan, Yuliang Liu, and Xiang Bai. Video streaming thinking: Videollms can watch and think simultaneously. In *European Conference on Computer Vision*, pp. 80–99. Springer, 2026.

Ilker Kesen, Andrea Pedrotti, Mustafa Dogan, Michele Cafagna, Emre Can Acikgoz, Letitia Parcalabescu, Iacer Calixto, Anette Frank, Albert Gatt, Aykut Erdem, et al. Vilma: A zero-shot benchmark for linguistic and temporal grounding in video-language models. In *International Conference on Learning Representations*, volume 2024, pp. 40504–40554, 2024.

Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 780–787, 2014.

Pengyiang Liu, Zhongyue Shi, Hongye Hao, and Qi Fu. Xueting bi, siwei zhang, xiaoyang hu, zitian wang, linjiang huang, and si liu. svcbench: A streaming video counting benchmark for spatial-temporal state maintenance. In *European Conference on Computer Vision*, volume 2, pp. 36, 2026.

Viorica Patraucean, Lucas Smaira, Ankush Gupta, Adria Recasens, Larisa Markeeva, Dylan Banarse, Skanda Koppula, Mateusz Malinowski, Yi Yang, Carl Doersch, et al. Perception test: A diagnostic benchmark for multimodal video models. *Advances in Neural Information Processing Systems*, 36:42748–42761, 2023.

Rohith Peddi, Shivvrat Arya, Bharath Challa, Likhitha Pallapothula, Akshay Vyas, Bhavya Gouripeddi, Qifan Zhang, Jikai Wang, Vasundhara Komaragiri, Eric Ragan, et al. Captaincook4d: A dataset for understanding errors in procedural activities. *Advances in Neural Information Processing Systems*, 37:135626–135679, 2024.

Zihan Qiu, Zekun Wang, Xiao Li, Yanpeng Li, Yang Xu, Yixuan Wang, Huaqing Zhang, Rui Men, Bochao Mao, Chengruidong Zhang, et al. On the design of qwen3. 8-next architecture: Evaluation, efficiency, and training stability. *arXiv preprint arXiv:2608.30320*, 2026.

Yansong Shi, Qingsong Zhao, Tianxiang Jiang, Xiangyu Zeng, Yi Wang, and Limin Wang. River: A real-time interaction benchmark for video llms. *arXiv preprint arXiv:2603.03985*, 2026.

Jawad Tayyub, Majd Hawasly, and Anthony G Cohn. Complex and long activities dataset. 2017.

Pengyu Wang, Chenkun Tan, Shaojun Zhou, Qirui Zhou, Yanxin Chen, Xingyang He, Huazheng Zeng, Jijun Cheng, Chenghao Wang, Xiaomeng Qian, et al. Moss-vl technical report. *arXiv preprint arXiv:2608.15045*, 2026a.

Yuan Wang, Shujian Gao, Songtao Jiang, Zhengyu Hu, and Zuozhu Liu. Medstreambench: A time-aware benchmark for streaming and proactive medical video understanding. *arXiv preprint arXiv:2607.01751*, 2026b.

Yueqian Wang, Songxiang Liu, Disong Wang, Nuo Xu, Guanglu Wan, Huishuai Zhang, and Dongyan Zhao. Mmduet2: Enhancing proactive interaction of video mllms with multi-turn reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 23257–23270, 2026c.

Spencer Whitehead, Suzanne Petryk, Vedaad Shakib, Joseph Gonzalez, Trevor Darrell, Anna Rohrbach, and Marcus Rohrbach. Reliable visual question answering: Abstain rather than answer incorrectly. In *European conference on computer vision*, pp. 148–166. Springer, 2022.

Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. *Advances in Neural Information Processing Systems*, 37:28828–28857, 2024.

Shuhang Xun, Sicheng Tao, Jungang Li, Yibo Shi, Zhixin Lin, Zhanhui Zhu, Yibo Yan, Hanqian Li, Linghao Zhang, Shikang Wang, et al. Rtv-bench: Benchmarking mllm continuous perception, understanding and reasoning through real-time video. *Advances in Neural Information Processing Systems*, 38, 2026.

Ziang Yan, Sheng Xia, Jiashuo Yu, Yue Wu, Tianxiang Jiang, Songze Li, Kanghui Tian, Yicheng Xu, Yinan He, Kai Chen, et al. Internvideo3: Agentify foundation models with multimodal contextual reasoning. *arXiv preprint arXiv:2606.12195*, 2026.

Zhenyu Yang, Yuhang Hu, Zemin Du, Dizhan Xue, Shengsheng Qian, Jiahong Wu, Fan Yang, Weiming Dong, and Changsheng Xu. Svbench: A benchmark with temporal multi-turn dialogues for streaming video understanding. In *International Conference on Learning Representations*, volume 2025, pp. 42522–42556, 2025.

Linli Yao, Yicheng Li, Yuancheng Wei, Lei Li, Shuhuai Ren, Yuanxin Liu, Kun Ouyang, Lean Wang, Shicheng Li, Sida Li, et al. Timechat-online: 80% visual tokens are naturally redundant in streaming videos. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 10807–10816, 2025.

Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, and Xiaojie Jin. Flash-vstream: Efficient real-time understanding for long video streams. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21059–21069. IEEE, 2025.

A Inference Prompts

Shared system instruction. MOSS-VL-Realtime, TimeChatOnline, VST, ReKV, LiveCC, Flash-VStream, and the asynchronous sliding-window protocol use the following system text.

```
You analyze a live video frame by frame. Follow the user's streaming response
    protocol exactly. Use <|silence|> whenever no new warranted response is
    required. Never use future information.
```

Initial observation instruction. Before the task is disclosed, the models receive:

```
Observe the incoming live video silently. A maintenance task will be issued later.
```

Complete task message. The following task becomes active at 6 s and remains active through the exclusive window end at 30 s.

```
Maintain one atomic fact for the rest of this live video.

Question: Continuously maintain whether the transparent pitcher with the orange lid
    and handle is resting on the wooden tabletop.
Target: The tall transparent pitcher with a matching orange lid and orange handle.
Scope: Maintain the same pitcher's tabletop-resting state from query issuance until
    the public window ends. Establish its identity when it becomes identifiable.
    Retain a supported resting state through ordinary occlusion, but reassess
    after manipulation that could lift or relocate it.
Decision rule: Base support by the wooden tabletop determines on_table; otherwise
    the physical state is off_table. Hand contact alone has no priority over
    established tabletop support. Unresolved support is grounds for deferral, not
    another value.
Visibility rule: Use the visible base, tabletop plane, vertical separation, and
    surrounding handling evidence. Disappearance does not establish off_table.
    Defer before sufficient target evidence or when an obscured manipulation
    leaves support unresolved; update or correct the same claim when evidence
    resolves it.

Allowed labels (copy exactly one): on_table, off_table
Definitions:
- on_table: The pitcher's base rests on the wooden tabletop. A hand touching its
    lid or handle does not by itself change this condition.
- off_table: The pitcher is lifted clear of the wooden tabletop or rests somewhere
    other than that tabletop.

Streaming response protocol:
1. If no label is sufficiently supported yet, output <|silence|>.
2. As soon as a label is sufficiently supported, output exactly that label and
    nothing else.
3. If later evidence supports a different label, output exactly the new label and
    nothing else.
4. If a prior commitment becomes unsupported before a replacement is justified,
    output exactly __DEFER__.
5. If the maintained state has not changed, output <|silence|>.
Do not explain or predict future frames.
```

B Asynchronous Sliding-Window Protocol

This protocol is used for Qwen3.8-Flash-Next, GLM-5.3-Flash, dots3-note-prev, MiMo-V2.6-Flash-RL, and InternVideo3-8B-Instruct.

The frozen runner plays each video at 1 Hz and schedules a request at video seconds $q + 4k < e$, where q is the query time and e is the exclusive window end. Each request contains the standing task, the latest at most 16 arrived frames in chronological order, their video times, and the model's latest applied public state. Requests are launched even when earlier requests remain in flight, so playback never waits for generation.

The request ends with the following instruction:

```
Your own latest available maintained state (not ground truth): {feedback_json}

The latest available video second is {current_second}. Continue the SAME
    maintenance task; moving the visual window does not reset your commitments.
    Keep, update, or retract them according to the available evidence and the
    streaming response protocol. Output only an allowed label, __DEFER__, or
    <|silence|>.
```

Before any valid response has been applied, the feedback instead states that the maintained state is `__DEFER__`. No reference annotation is supplied.

Responses take effect when generation completes. Completion time is mapped to video time and rounded down to an integer second. A response is applied only inside $[q, e)$ and only if its request uses a newer source tick than the latest applied response; therefore, a delayed older request cannot overwrite a newer state. A valid label updates the public state, `__DEFER__` withdraws it, and `<|silence|>` preserves it. Invalid and late responses are logged without replacing the state. If several valid responses apply in the same second, the last one is retained for that second.

Frames exceeding 262,144 pixels are resized with aspect ratio preserved and encoded as JPEG at quality 90. The frozen defaults are a 16-frame window, a 4 s request stride, at most 128 output tokens per request, and four concurrent tasks. The runner records frame times, request and completion times, outputs, errors, and skipped or delayed scheduling slots.

C Annotation Pipeline Summary

The offline construction pipeline is separate from inference. Its stages use independent contexts and receive only the public task, permitted upstream candidates, and ordered frames. They do not receive human labels or evaluation scores. Table 6 summarizes the five functional stages; complete frozen prompts, schemas, and instantiated requests are supplied with the accompanying artifact.

Table 6: Functional stages of the annotation pipeline.

Stage	Function
Task design and review	Drafts an atomic task, checks its target, labels, scope, and evidence rules, and repairs or rejects invalid candidates.
Local and global analysis	Combines overlapping 32-frame clips with a sparse whole-video scan to find the target and possible state changes.
Timeline drafting	Constructs a provisional sequence of states and the earliest evidence supporting each answer.
Refinement and review	Rechecks all sampled frames, both sides of boundaries, short-lived states, counts, and disagreements.
Validation and freezing	Enforces structural consistency and selects the first valid candidate, using a valid upstream result as fallback when necessary.

The resulting annotation distinguishes when a state holds from when the available video prefix supports reporting it. Deterministic validation converts the selected annotation into the reference trajectory used for evaluation. Two authors then review every retained task against its video and task definition; tasks with incorrect reference annotations are discarded.

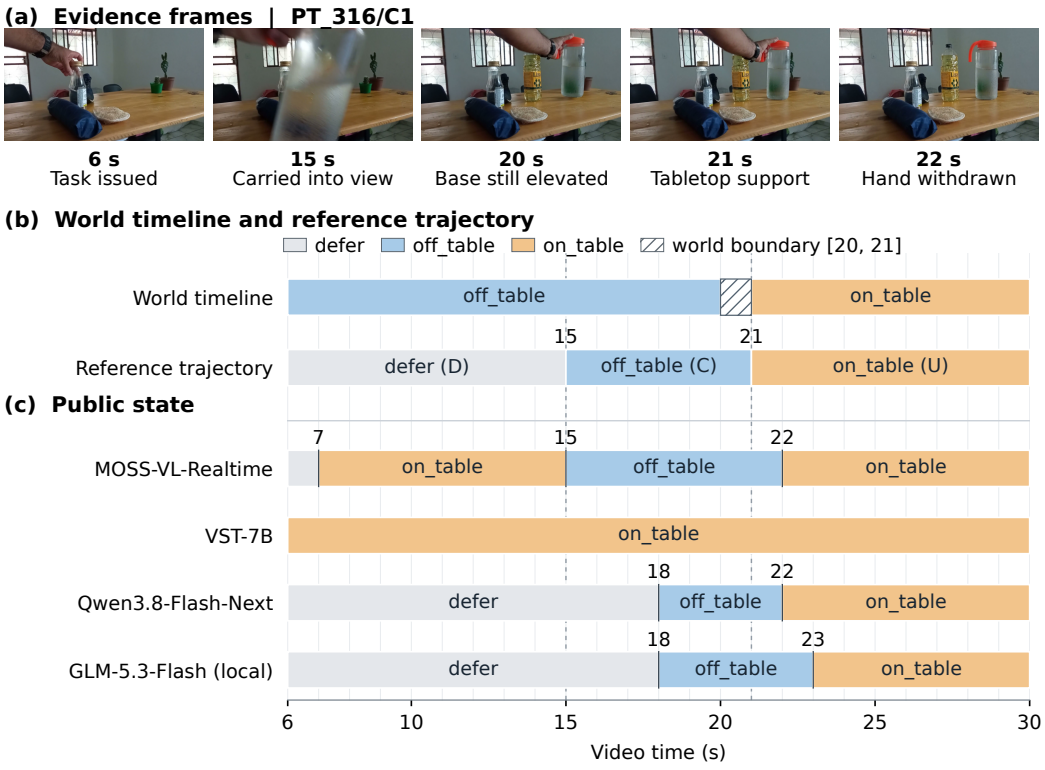


Figure 4: **Recorded trajectories for the pitcher task (`PT_316/C1`).** (a) Sampled video frames show the evidence available over time. (b) The reference trajectory defers on $[6, 15)$, commits to `off_table` on $[15, 21)$, and updates to `on_table` on $[21, 30)$. (c) Four models' saved public states are plotted at the integer seconds when their outputs take effect. Gray, blue, and orange denote defer, `off_table`, and `on_table`, respectively. These recorded outputs illustrate individual behaviors rather than aggregate performance.

D A Recorded Answer-Maintenance Trajectory

Figure 4 shows a recorded instance of the pitcher task used in Figure 1. The task, `PT_316/C1`, asks whether the transparent pitcher with the orange lid and handle rests on the wooden tabletop. The allowed labels are `on_table` and `off_table`, and evaluation covers integer seconds $[6, 30)$.

The pitcher is initially off the table, but the available frames do not support identifying it at the query time. The reference therefore remains defer until 15 s, when the pitcher becomes identifiable in the foreground. Visible clearance remains beneath its base at 20 s; at 21 s, the base reaches the tabletop while the hand still touches the handle. The supported answer therefore changes from `off_table` to `on_table` at 21 s.

The four trajectories illustrate distinct failure patterns. MOSS-VL-Realtime answers `on_table` prematurely at 7 s, corrects to `off_table` when evidence becomes sufficient at 15 s, and updates to `on_table` at 22 s. VST-7B maintains `on_table` throughout the window. Qwen3.8-Flash-Next and GLM-5.3-Flash remain deferred until 18 s, three seconds after the initial supported commitment, and update to `on_table` at 22 s and 23 s, respectively. The official ± 1 s tolerance changes which states are accepted for scoring; it does not move the reference boundaries or the recorded model transitions.