PR #27249 完整报告

sgl-project/sglang

[diffusion] Fix realtime webui recording timeline

合并时间：2026-06-04 19:58

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27249

执行摘要

此 PR 修复了 realtime WebUI 录制功能中 MP4 输出与预览帧序列不一致的问题。通过将帧捕获点从画布后移至解码后，并使用固定 FPS 时间线替代实时时间戳，使录制视频与生成的帧序列保持一致，不再受预览丢帧和延迟影响。

功能与动机

实时 WebUI 的录制功能之前依赖于从画布捕获帧，但画布帧受预览播放器的帧率控制和丢帧策略影响，导致录制的 MP4 视频中出现卡顿和跳帧。PR body 明确指出："Record decoded source frames before realtime preview playback can hold or drop frames"，需要让录制直接从解码后的帧序列中采样，避免预览缓冲的干扰。

实现拆解

1. 新增录制专用画布和帧时间线：在 app.js 中创建独立的 recordingCanvas 和 recordingCtx，用于处理 ImageData 类型的帧源。用 recordingFps（固定 FPS）替代原来的 recordingStartedAt 和 recordingFirstFrameAt，时间戳改为基于 frameIndex × fiexed_duration，不再依赖 performance.now()。
2. 重构帧捕获入口：将 captureRecordingFrame(item, now) 替换为 recordDecodedFrame(image)，该函数直接从解码后的帧（item.image）创建 VideoFrame，使用 createRecordingFrame 将 ImageData 绘制到专用画布上。新增 recordDecodedFrameBatch(decodedFrames) 处理批量帧录入。
3. 更新 UI 和初始化逻辑：updateRecordButton 中录制时长改为 recordingFrameIndex / recordingFps，更精确反映已记录的帧数。startRecording 时固定 recordingFps 为当前预览目标 FPS。
4. HTML 缓存版本更新：index.html 中 app.js 的缓存版本从 v72 改为 v73。

python/sglang/multimodal_gen/apps/realtime_webui/app.js

核心变更文件，重构了录制帧捕获逻辑，新增画布和FPS固定时间线，是功能修复的主要实现。

```
// 新增专用录制画布，用于从 ImageData 源创建 VideoFrame
const recordingCanvas = document.createElement("canvas");
const recordingCtx = recordingCanvas.getContext("2d", { alpha: false });

// 录制帧索引和固定 FPS ( 在 startRecording 时从预览目标 FPS 锁定 )
let recordingFrameIndex = 0;
```

```
let recordingFps = DEFAULT_TARGET_FPS;

// 从解码帧批量录入
function recordDecodedFrameBatch(decodedFrames) {
  if (!recordingActive || recordingSaving) return;
  for (const item of decodedFrames) {
    if (!recordingActive) break; // 录制中途停止则退出
    recordDecodedFrame(item.image); // 录入每帧解码后的图像
  }
  updateRecordButton();
}

// 录入单个解码帧
function recordDecodedFrame(image) {
  if (!recordingActive || recordingSaving) return;
  const frameIndex = recordingFrameIndex;
  // 基于固定 FPS 计算每帧时长（微秒）
  const duration = Math.round(1_000_000 / Math.max(1, recordingFps));
  const timestamp = frameIndex * duration; // 帧索引 × 固定帧时长
  let frame;
  try {
    // 使用 createRecordingFrame 将 ImageData 绘制到专用画布并创建 VideoFrame
    frame = createRecordingFrame(image, timestamp, duration);
  } catch (error) {
    recordingActive = false;
    addHistory(error.message || "recording frame capture failed");
    updateRecordButton();
    return;
  }
  recordingFrameIndex += 1;
  recordingEncodeChain = recordingEncodeChain
    .then(async () => {
      await ensureRecordingEncoder(frame.displayWidth, frame.displayHeight);
      recordingEncoder.encode(frame, { keyFrame: frameIndex === 0 || frameIndex % 120 === 0
      });
      frame.close();
    })
    .catch((error) => {
      // 注意：error 发生时会停止录制，但不会阻止后续帧执行 .then，可能导致级联错误
      recordingActive = false;
      addHistory(error.message || "recording encode failed");
      updateRecordButton();
    });
}
```

评论区精华

> 当编码出错时，仅设置 recordingActive = false，但后续帧仍会执行 .then，导致每个剩余帧都触发错误，日志淹没。建议在 .then 块开头检查 !recordingActive &&

> recordingSaving 以跳过编码。 —— gemini-code-assist[bot]，review 评论

该问题在 PR 中未得到解决，属于已识别的改进点。

风险与影响

- 风险：编码错误可能引发级联失败，导致大量错误日志；recordingFps 在录制开始后固定，若用户动态调整 FPS 不会影响录制。缺少自动化测试，回归风险无法通过 CI 捕获。
- 影响：仅影响 realtime WebUI 的录制功能，录制的 MP4 文件更流畅，时长显示更精确。对系统性能影响轻微。

关联脉络

与 #27236（加速 RGB 传输）同属 diffusion 模块的实时流功能改进，共同提升实时视频体验。
也与#26119（diffusion解耦参数与预热工具）有间接关联，后者为此类特性提供了基础设施。