基于Agent架构的AIGC短视频智能创作系统设计与实践

张 伟

（上海哔哩哔哩科技有限公司，上海 200433）

摘 要： 针对短视频工业化创作中交互效率低、素材覆盖不足、高并发调度困难等技术问题，提出一种基于Agent架构的TTV（Text-to-Video）智能创作系统。该系统采用图形用户界面（GUI）与语言用户界面（LUI）双模交互架构，融合大语言模型（LLM）分镜生成、多源素材检索、TTS语音合成与视频自动合成等技术，实现从文本到视频的端到端自动化创作。重点阐述3项关键技术的设计与实现：（1）基于Skills，subagent和tool call三层机制的Agent双模交互架构；（2）面向千万级素材的三层检索匹配策略；（3）基于表隔离与消息缓冲的并发排队调度机制。经生产环境验证，系统素材库规模达到千万级，创作周期由数天缩短至分钟级，批量编辑效率提升20倍以上，为AIGC技术在数字媒体领域的工业化落地提供了工程实践参考。

关键词： AIGC；AI Agent；TTV；短视频创作；素材智能检索

中图分类号： TP311 **文献标识码：** A

Design and practice of AIGC short video intelligent creation system based on Agent architecture

ZHANG Wei

（Shanghai Bilibili Technology Co.，Ltd.，Shanghai 200433，China）

Abstract: To address the technical challenges of low interaction efficiency, insufficient material coverage, and difficulties in high-concurrency scheduling in industrialized short video creation, an intelligent TTV (Text-to-Video) creation system based on an Agent architecture is proposed. A dual-mode interaction architecture combining Graphical User Interface (GUI) and Language User Interface (LUI) is adopted, and technologies including Large Language Model (LLM)-based shot generation, multi-source material retrieval, TTS speech synthesis, and automatic video composition are integrated to achieve end-to-end automated creation from text to video. Three key technologies are mainly described: an Agent dual-mode interaction architecture based on a three-layer mechanism comprising Skills, subagents, and tool calls, a three-layer retrieval and matching strategy for tens of millions of materials, and a concurrent queuing and scheduling mechanism based on table isolation and message buffering. Verified in a production environment, the material library of the system reaches a scale of tens of millions, the creation cycle is reduced from several days to several minutes, and batch editing efficiency is improved by more than 20 times, thereby providing an engineering reference for the industrial application of AIGC technology in the digital media field.

Key words: AIGC, AI Agent, TTV, short video creation, intelligent material retrieval

短视频作为互联网内容传播的主要形式，国内头部平台日活用户已达到数亿规模。然而，短视频创作仍面临效率和规模化方面的瓶颈。传统模式下一条视频通常需要经历创意构思、脚本撰写、素材搜集、剪辑合成等环节，制作周期较长、人工成本较高。随着LLM等AIGC技术不断发展，短视频创作迎来了自动化变革的机遇[1]。然而，如何将AIGC技术进一步转化为稳定、高效的工业化创作系统，仍面临3个核心问题。

（1）交互效率瓶颈：现有工具多依赖GUI交互，用户需要逐个调整分镜对应的素材和文案，当视频包含数十个分镜时，批量编辑效率较低。

（2）素材覆盖不足：单一来源的素材库覆盖范围有限，且存在长尾类别缺失、更新能力不足等问题，难以满足多样化的短视频创作需求。

（3）高并发调度困难：视频生成涉及LLM调用、TTS合成、素材检索等资源消耗较高的环节，在大量用户并发创作的情况下，容易产生资源争抢，影响系统稳定运行[2]。

针对上述问题，本文以TTV智能创作系统为载体，围绕Agent双模交互架构、多源素材三层检索匹配和并

发排队调度优化3项关键技术展开论述。

1　TTV 系统整体架构设计

1.1　系统核心流程

TTV 系统将短视频制作模式由“人工主导”转变为“AI辅助+人工精调”。其整体创作流程分为7个阶段：用户输入文稿或上传音频；LLM 进行语义分析，自动拆分分镜并生成画面描述；系统根据画面描述从多源素材库中智能检索并匹配视频素材；TTS 引擎生成配音及字幕；系统自动完成素材、配音和字幕的合成；用户通过 GUI 或 Agent 对话进行精细化编辑；导出或发布视频。

1.2　技术架构分层

如图1所示，系统采用四层架构。其中，交互层支持 GUI 可视化操作和 Agent LUI 对话两种交互模式。业务服务层采用 Golang 开发，通过 RESTful API 承担项目管理、分镜 CRUD、排队调度等核心业务逻辑[3]。Agent 智能层采用 Python 构建，内部划分为 Skills 能力管理、subagent 编排和 tool call 工具调用3个子层，并通过 SSE（Server-Sent Events）流式协议与业务服务层进行通信。数据与存储层采用多引擎组合方案：MySQL 用于存储项目、段落、分镜等结构化数据；分布式 KV 存储用于承载分镜候选列表，key 格式为 candidate_{clipId}_{序号}；Redis 用于缓存限流计数和排队状态；消息队列用于实现异步削峰。

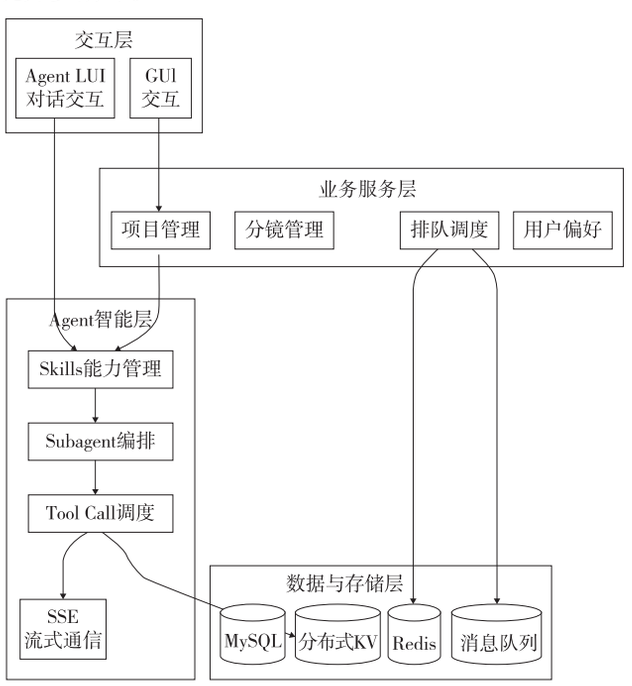

图1　TTV 系统四层技术架构

1.3　数据模型设计

如图2所示，系统核心数据模型采用三级结构，包括项目（Project）、段落（Part）和分镜（Clip）。项目表主要存储用户 ID、脚本文本、阶段状态、版本号等字段，并通过 priority 字段支持调度优先级设置；段落表主要记录排序索引和内容文本；分镜作为最小编辑单元，包含 content（分镜文本）、screen（画面描述）、video_candidates（候选素材集）、version（版本号）、index_float（浮点排序索引）、clip_from（素材来源标识，其中0表示素材库、1表示本地素材）、start_time（切片分段起始时间）等关键字段。

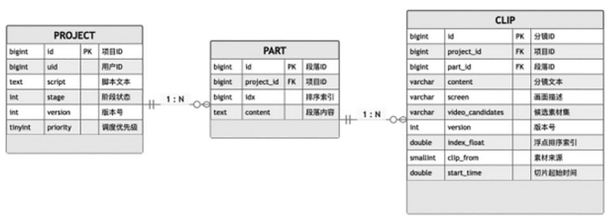

图2　三级数据模型关系

2　关键技术方案与实现

2.1　基于 Agent 的双模交互架构

2.1.1　架构设计

针对短视频创作过程中的交互效率瓶颈，系统设计了“GUI+LUI”双模交互的 Agent 架构（见图3）。Agent 服务基于 Python agno 框架独立部署，通过 Skills 管理、subagent 编排和 tool call 调用三层机制协同完成创作任务[4-5]。

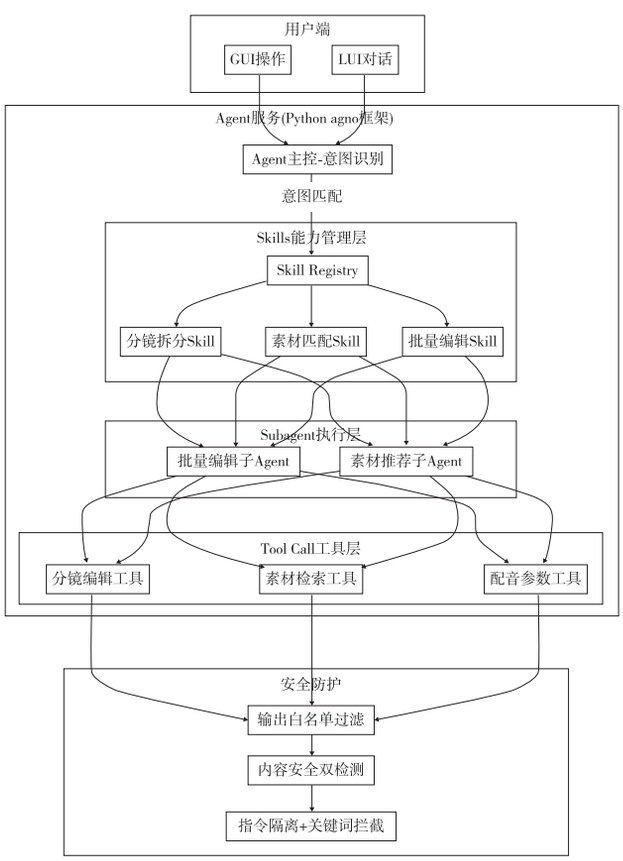

图3　Agent 双模交互架构

Skills 模块是 Agent 的核心能力抽象层。每个 Skill 定义一类完整的创作任务处理流程，包括任务识别规则、执行步骤编排、领域提示词模板和输出格式约束。系统预置了分镜智能拆分 Skill、素材风格匹配 Skill 和批

量编辑Skill等核心能力。其中,分镜智能拆分Skill负责编排TTS切割和素材重搜流程;素材风格匹配Skill封装从画面描述生成检索query到候选素材排序的完整链路;批量编辑Skill则将自然语言指令解析为针对多个分镜的并行操作计划。

Skills采用注册式管理方式,通过Skill Registry统一完成Skill的注册与发现。Agent主控接收用户指令后,首先进行意图识别并匹配对应Skill,再由Skill编排subagent和tool call的执行序列。其伪代码逻辑如下。

算法1 Agent指令处理流程

输入:user_input(用户输入),project_context(项目上下文)

输出:execution_result(执行结果)

```
matched_skill = SkillRegistry.match(user_input)
if matched_skill == null then
    return LLM.generate_response(user_input)
end if
task_plan = matched_skill. decompose (user_input, project_context)
for each sub_task in task_plan do
    if sub_task.is_parallel then
        subagent = SubAgentPool.acquire()
        subagent.execute_async(sub_task.tool_calls)
    else
        ToolExecutor.execute(sub_task.tool_calls)
    end if
    SSE.push_event(sub_task.progress)
end for
return aggregate_results(task_plan)
```

在底层工具层面,系统将分镜文本编辑、素材检索替换、画面风格调整、配音参数修改等原子操作分别封装为独立工具。对于“将所有户外分镜替换为夜景风格并降低语速”等复杂指令,Skill将任务进一步拆解为分镜筛选、素材检索、风格替换和配音调整等子任务,并通过subagent并行执行,从而提高复杂任务的处理效率。

在交互协议方面,用户通过/chat/run接口发起多模态对话,支持文本和图片输入,系统返回唯一的run_id,随后通过SSE流式接口持续推送任务执行事件。同一项目下,LUI仅允许一个Run处于执行状态,并通过is_running标识避免并发冲突。同时,系统提供创作偏好管理功能,用户可以预设风格偏好,Skill执行任务时自动加载相关偏好配置。

2.1.2 三层安全防护设计

Agent交互面临与传统编辑工具不同的对话安全风险。针对这一问题,系统设计了三层安全防护机制。

(1)输出白名单过滤。Agent执行过程中产生的中间状态、调试信息和内部调用链路等内容,通过预定义的白名单正则规则集进行拦截[6]。白名单规则采用配置文件统一管理,仅允许输出进度提示、最终结果等符合规则的消息,不符合白名单规则的输出一律过滤,以降低内部实现细节泄露风险。

(2)内容安全双重检测。系统对Agent输入和输出均接入平台内容安全服务,采用规则引擎与机器学习分类模型进行双重检测。其中,规则引擎通过正则关键词库覆盖不良内容类别,处理速度达到毫秒级;机器学习模型则进一步结合上下文语义进行判定。当任一检测机制触发风险规则时,系统即对相关内容进行拦截。

(3)Skill能力防泄露。针对用户可能通过Prompt注入探取Skill配置和system prompt的风险,系统在Agent框架层实现指令隔离,将系统级指令(Skill定义、编排逻辑)与用户级对话内容在LLM上下文窗口中通过system,user,assistant等角色标签进行严格区分。同时,对Agent输出执行关键词过滤,检测并拦截包含Skill名称、内部工具标识和编排流程描述等敏感信息的响应。

2.2 多源素材智能检索与匹配

2.2.1 三层素材体系设计

针对素材覆盖不足问题,系统设计了三层素材供给体系。

(1)平台稿件自动化切片。系统从平台日增投稿视频中按照设定条件进行筛选,日均可筛选稿件约25万条。筛选后的视频依次经过视频云无水印转码、AI算法自动切片、内容安全机审和版权送审等流程后自动入库,全流程无需人工干预。其中,视频云采用独立转码配额,720p素材P80转码耗时约1 min;内容安全机审则根据白天低峰、夜间高峰等时段进行差异化调度,以充分利用审核接口QPS配额。目前,历史已入库可用切片达到数百万条规模。

(2)PGC专业素材接入。系统接入第三方版权素材供应商,获得千万级专业素材。技术方案采用异步预处理方式,而非实时同步处理:标清素材(720p)经过预处理后入库,供在线创作使用;高清素材(1 080p)则在视频导出阶段按需获取。针对千万级存量素材,系统按照时长分布进行分批导入,其中约九成素材时长在30 s以内,转码耗时较短,因此优先处理短时长素材,以快速扩大素材覆盖范围。在导出阶段,若检测到分镜使用第三方素材且本地不存在高清版本,系统自动创建包含下载、转码和切片操作的异步任务。

(3)用户本地素材上传。系统支持用户上传个性化视频和图片素材,如片头、转场等。上传素材经转码和机审后即可用于分镜编辑。系统采用独立的用户素材表,并通过clip_from字段进行来源区分,其中值为1表示用户本地素材,从而实现用户素材与公共素材库的隔离管理。

2.2.2 实时检索补充策略

在三层素材供给体系基础上,系统进一步设计了实

时检索补充机制，以解决长尾类别素材不足的问题。当某一类别的素材数量低于可配置的充足阈值时，系统根据分镜画面描述自动生成检索 query，并调用平台搜索接口获取 Top-N 稿件，默认 N=50。经过过滤后，选取 Top-5 结果推送至切片入库链路。

素材表中设置 priority 字段，用于区分不同来源素材的优先级。其中，实时检索素材的 priority 设置为 4，高于日常增量素材的默认值 0，从而确保即时补充的素材能够优先参与匹配。检索结果通过 DataBus 消息异步推送，消息主题为 Ttv-Archive-Query-T，消息体包含 project_id、clip_id、画面描述等字段，由下游消费者接收消息并触发后续切片流程。

2.3　并发排队调度优化

2.3.1　三级限流策略设计

针对短视频创作过程中存在的高并发调度问题，系统设计了三级递进式限流策略（见图 4）。限流参数通过配置中心 Paladin 进行动态管理，各级阈值均可根据系统运行情况实时调整。

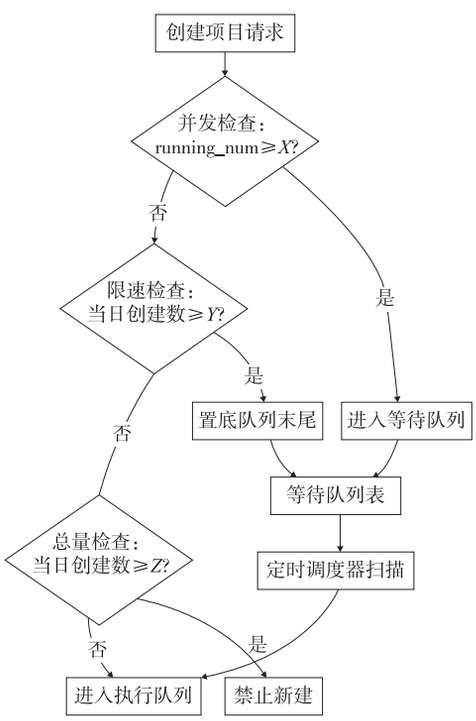

图 4　三级限流策略

第一级为并发限制，限制每个账号同时生成的项目数上限 X，默认值为 2。当同时生成的项目数超过该阈值时，新项目进入等待队列。第二级为限速机制，当单日创建项目数超过阈值 Y，默认值为 6 时，新创建项目进入队列末尾，并将 priority 设置为 -1，以降低调度优先级。第三级为总量上限，当单日创建项目数达到上限 Z，默认值为 50 时，系统禁止继续创建新项目，以实现系统过载保护。

2.3.2　技术实现

排队调度的技术实现主要包含表隔离、分布式限流、消息缓冲削峰和排队时间预估 4 项关键机制。

（1）表隔离设计：系统新增独立的 project_wait 等待队列表，与执行中的 projects 表进行物理隔离。等待队列表包含 uuid，uid，script，voice_id，priority，state 等字段，其中 state=-2 表示排队中，-1 表示已取消，0 表示待执行。定时调度器按照 priority 降序、ctime 升序的顺序扫描等待队列，并将符合条件的项目推入执行队列。

（2）分布式限流：系统采用 Redis 实现用户维度的限流计数。其中，并发数通过 key running_pro_num：{mid} 进行记录，TTL 设置为 1.5 天；项目创建频率通过 key pro_create_rate：{mid} 进行记录。限流检查的核心逻辑如下。

算法 2 Redis 分布式限流检查

```
输入：mid（用户 ID），config（限流配置）
输出：action（允许/排队/置底/拒绝）
running_count = Redis.GET("running_pro_num:{mid}")
if running_count >= config.running_num_max then
    return QUEUE
end if
daily_count = Redis.GET("pro_create_rate:{mid}")
if daily_count >= config.project_create_max_day then
    return REJECT
end if
if daily_count >= config.high_priority_count then
    return LOW_PRIORITY
end if
Redis.INCR("running_pro_num:{mid}")
Redis.INCR("pro_create_rate:{mid}")
return ALLOW
```

（3）消息缓冲削峰：项目创建请求首先写入消息队列，消息主题为 Ttv-Project-Create-T，再由消费者按照可控速率消费并写入数据库，从而避免大量请求瞬时写入数据库造成过高压力。

（4）排队时间预估：系统根据当前队列深度、近期平均生成时长和系统并发处理能力，实时计算项目预计等待时间，并通过 /project/info 接口的 wait_time 字段返回客户端，单位为 s，其中 -1 表示当前无需排队。计算公式为：

$$T_{\text{wait}}=\frac{Q_{\text{depth}}}{C_{\text{parallel}}}\times T_{\text{gen}} \tag{1}$$

其中，Q_{depth} 为当前用户前方的排队项目数，C_{parallel} 为系统并发处理能力，即系统同时可执行的项目数；T_{gen} 为近期项目的平均生成时长。该机制能够使用户在项目创建后及时获知预计等待时间。

3　系统验证与效果分析

3.1　Agent 交互效率验证

Agent 双模交互架构上线后，以“调整 20 个分镜的素

材风格”为基准进行对比测试，测试覆盖批量风格替换、批量文案修改、配音参数调整和指定类别分镜筛选与替换4类典型场景。测试结果如表1所列。

表1 Agent交互效率对比数据

操作场景	GUI模式耗时	LUI模式耗时	效率提升倍数
批量风格替换	10~15 min	约30 s	20~30倍
批量文案修改	8~10 min	约20 s	24~30倍
配音参数调整	5~8 min	约10 s	30~48倍
指定类别分镜筛选与替换	12~18 min	约40 s	18~27倍

三层安全防护机制经专项测试得到验证：输出白名单过滤能够拦截测试过程中产生的中间状态等内部信息；内容安全双重检测通过规则引擎与ML模型协同覆盖多类敏感内容；在Prompt注入攻击测试中，指令隔离和关键词过滤机制均未出现Skill信息泄露，能够满足Agent交互过程中的基本安全防护需求。

3.2 素材覆盖率验证

三层素材体系部署后，系统素材库规模由初期依靠人工入库的数十万条扩展至千万级。各层素材在存量规模、增量方式、转码处理和高清素材获取等方面的量化情况如表2所列。

表2 三层素材体系量化数据

指标	第一层（平台切片）	第二层（PGC素材）	第三层（用户素材）
存量规模	数百万条	千万级	按用户独立管理
日均增量	约25万条可筛选稿件	按批次导入	用户实时上传
720p转码P80耗时	约1 min	约1 min	约1 min
入库链路延迟	全自动，分钟级	异步预处理，零在线延迟	“转码+机审”后即可使用
1 080p获取策略	直接入库	导出时按需获取	用户原始上传

PGC素材采用异步预处理方案后，相较于实时处理方案，每个分镜的在线处理延迟减少约3~4 min。实时方案需要依次完成下载、上传、转码、机审和切片等流程，而异步预处理将相关操作前置，从而减少高清素材处理对在线创作流程造成的影响。

此外，千万级存量素材中约九成时长在30 s以内。系统优先导入短时长素材，使素材库能够在较短时间内快速扩大基础覆盖范围。针对长尾类别素材不足的问题，实时检索补充机制根据分镜画面描述动态生成检索query，并通过priority字段提高即时补充素材的匹配优先级，从而进一步增强素材供给能力。

3.3 并发调度稳定性验证

三级限流策略和表隔离设计部署后，系统能够在高并发创作场景下保持稳定运行。排队调度系统的主要运行参数和技术机制如表3所列。

表3 排队调度系统运行数据

指标	数据
并发限制默认值（X）	2个项目/账号
限速阈值默认值（Y）	6个项目/天/账号
总量上限默认值（Z）	50个项目/天/账号
等待队列与执行队列	物理表隔离，互不干扰
消息缓冲	DataBus异步削峰，数据库写入平滑
Redis限流键TTL	1.5天（running_pro_num）
排队预估时间误差	可接受范围内

在高并发场景下，等待队列表与执行队列采用物理表隔离方式，使排队项目与生成中项目相互独立，避免排队逻辑对执行中项目产生额外性能干扰。消息缓冲机制将项目创建请求通过DataBus异步处理，使数据库写入由瞬时峰值转变为相对平稳的持续写入，从而降低突发请求对数据库造成的压力。同时，排队预估算法根据队列深度、系统并发处理能力和近期平均生成时长实时计算预计等待时间 $T_{\text{wait}}=\frac{Q_{\text{depth}}}{C_{\text{parallel}}}\times T_{\text{gen}}$。预估结果通过接口返回客户端，使用户能够在等待过程中获得较为明确的时间预期，改善排队状态下的交互体验。

3.4 分镜编辑能力验证

在分镜编辑功能验证中，系统支持用户在文本任意换行位置执行分镜拆分，并根据拆分点对应的SRT字幕时间戳，对TTS音频波形和字幕序列进行精确切割。拆分后，前分镜沿用原素材，后分镜触发素材重新检索，同时执行浮点索引重排和版本号冲突校验，从而保证分镜拆分后数据关系和编辑顺序的一致性。

在版本控制方面，拆分请求携带客户端clip_version，服务端首先进行版本一致性校验，校验通过后执行拆分操作；若版本不一致，则返回错误码10006，并提示用户刷新后重试。该机制能够有效避免多端或并行编辑情况下产生的版本冲突。

切片分段功能通过clips表中的start_time字段记录素材选段的起始时间，用户可以根据创作需要精确选择素材的有效播放区间。当配音时长超过当前选段范围时，系统采用慢倍速播放方式进行时间拉伸，而不向前延伸素材起始点，从而保证用户选定的首帧画面保持不变。

在素材预览方面，系统采用按需截帧机制，用户打开预览时才触发截帧生成，而非对全量千万级素材进行预先截帧。该方式能够避免大规模素材预截帧带来的额外存储开销，同时满足用户在实际编辑过程中的预览需求。

4 结束语

针对短视频工业化创作过程中存在的交互效率低、

素材覆盖不足和高并发调度困难等问题，本文基于TTV智能创作系统提出了相应的系统化解决方案。通过构建由Skills，subagent和tool call组成的三层Agent架构，并结合SSE流式通信，实现GUI与LUI双模交互，使批量编辑效率提升20倍以上；通过平台稿件切片、PGC专业素材接入和用户本地素材上传构建三层素材体系，并结合实时检索补充机制，使素材库规模达到千万级；通过三级限流、表隔离和消息缓冲削峰等机制，实现对高并发创作请求的有序调度，保障系统稳定运行。

上述方案已在生产环境中得到验证，在提升短视频创作效率、扩大素材覆盖范围和增强并发处理能力等方面取得了较好的应用效果，可为AIGC技术在数字媒体领域的工业化落地提供一定的工程实践参考。

参考文献：

[1] LEWIS P, PEREZ E, PIKTUS A, et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks [C] // Advances in Neural Information Processing Systems, 2020: 9459-9474.

[2] VASWANI A, SHAZEER N, PARMAR N, et al. Attention Is All You Need[C]// Advances in Neural Information Processing Systems, 2017: 5998-6008.

[3] BROWN T, MANN B, RYDER N, et al. Language Models are Few-Shot Learners [C] // Advances in Neural Information Processing Systems, 2020: 1877-1901.

[4] ROMBACH R, BLATTMANN A, LORENZ D, et al. High-Resolution Image Synthesis with Latent Diffusion Models [C]// IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022: 10684-10695.

[5] RADFORD A, KIM J W, HALLACY C, et al. Learning Transferable Visual Models From Natural Language Supervision [C] // International Conference on Machine Learning, 2021: 8748-8763.

[6] GOODFELLOW I, POUGET-ABADIE J, MIRZA M, et al. Generative Adversarial Nets [C] // Advances in Neural Information Processing Systems, 2014: 2672-2680.

作者简介：

张伟（1990—），本科，高级工程师，研究方向：人工智能、计算机视觉、短视频智能创作，E-mail：ivan_cpp@126.com。

（上接第197页）

额外硬件逻辑器件支持的条件下，实现座舱显示系统仿真环境中的软件化视频叠加功能。

5　结束语

针对SCADE Display缺少直接视频处理能力、传统视频叠加功能依赖硬件逻辑器件实现的问题，本文结合OpenCV视频处理能力与SCADE Display纹理贴图机制，开展了软件化视频叠加功能实现研究；通过分析SCADE Display图形建模、纹理管理及显示更新机制，建立了基于视频帧解码、纹理资源替换和动态显示更新的视频叠加实现流程。后续可进一步结合视频编码优化、实时数据处理与嵌入式硬件平台验证等方向，从而提升该方法的工程应用能力。

参考文献：

[1] COLAÇO J L, CASPI P, HALBWACHS N. An overview of SCADE, a synchronous language for safety-critical software [J]. ACM Transactions on Embedded Computing Systems, 2020, 19(5): 1-25.

[2] BERRY G. Synchronous design and verification of critical embedded systems using SCADE and Esterel [C] // Formal Methods for Industrial Critical Systems: 12th International Workshop, FMICS 2007. Berlin: Springer, 2007: 1-8.

[3] 李虎，马晋，郑凤，等.SCADE模型驱动开发过程研究及高安全性分析[J].航空电子技术，2013(1)：15-19.

[4] JOSHI A, HEIMDAHL M P E. Model-based safety analysis of Simulink models using SCADE Design Verifier [C] // SAFECOMP 2005: Computer Safety, Reliability, and Security. Berlin: Springer, 2005: 122-135.

作者简介：

王文恺（1991—），硕士，工程师，研究方向：图形图像处理算法。