PR #34243 完整报告

sgl-project/sglang

[Diffusion] Serve Cosmos3 policies through the Action API

合并时间：2026-08-10 18:16

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34243

执行摘要

- 一句话：Cosmos3 动作模式迁移至通用 Action API
- 推荐动作：值得精读。三个看点：① protocol.py 的 Cosmos3 特化分支（capabilities 位 + 请求降级函数）展示了如何在统一 TI2V 管线上叠加通用 Action API；② Cosmos3DecodingStage 的 ACTION 短路设计避免动作请求为视频产物付费；③ vla→action 全局重命名对模块边界的整理。若你在维护对外 API 或计划接入新动作产出模型，可复用这套模式。

功能与动机

PR body 明确说明：Cosmos3 policy 和 inverse-dynamics 模式产生连续机器人动作，但此前通过异步 /v1/videos API 暴露，导致响应契约依赖视觉任务，且即使主要结果是动作张量，也会不必要地进入视频输出处理。因此需要将动作产出的模式迁移到 SGLang 通用 Action API，同时保留 Cosmos3 的视觉生成能力（forward_dynamics 仍走 /v1/videos）。

实现拆解

1. 端点能力抽象与路由：在 configs/pipeline_configs/base.py 新增 supports_action_endpoint() 与 supports_openpi_endpoint() 能力位；Cosmos3Config 覆盖 supports_action_endpoint，Pi05PipelineConfig 覆盖 supports_openpi_endpoint。http_server.py 据此挂载 action_api.router 与 openpi.router，替代原先基于 task_type.is_action_gen() 的判断。
2. Action 协议泛化与 Cosmos3 特化：runtime/entrypoints/vla/ 整体重命名为 runtime/entrypoints/action/；protocol.py 将采样参数类解析放宽为 SamplingParams | ActionSamplingParams，新增 _build_cosmos3_action_sampling_params 与 _cosmos3_image_from_observation，把图像 / 视频观测、action_horizon、domain_name 等参数下沉为 Cosmos3SamplingParams；action_metadata 对 Cosmos3Config 输出独立契约（modalities、supported_resolutions、action_modes、capabilities 等）。
3. 采样参数与解码行为：configs/sample/cosmos3.py 的 _adjust 根据 action_mode 校验并将 policy / inverse_dynamics 请求置为 DataType.ACTION，同时关闭 save_output、return_file_paths_only、return_frames 等视觉产物；Cosmos3DecodingStage.forward 在 DataType.ACTION 时直接返回动作张量并携带 domain_id、raw_action_dim 元数据，绕过 VAE decode 与 guardrails；Cosmos3LatentPreparationStage 也允许 ACTION 类

型复用图像 / 视频条件 latent。

4. 视频入口保护：runtime/entrypoints/openai/video_api.py 的 _build_video_sampling_params 在构造出 DataType.ACTION 的采样参数时抛 ValueError，由 HTTP 层转为 400，提示客户端改用 /v1/actions/generations。
5. 测试与文档配套：test/unit/test_cosmos3.py 新增 TestCosmos3ActionEndpoint（请求降级、metadata、响应格式、forward_dynamics 拒绝、action-only decode）与采样参数 _adjust 行为测试；docs/cookbook/diffusion/Cosmos/Cosmos3.mdx 更新为规范 Action API 示例与三种模式的路由说明。

关键文件：

- python/sglang/multimodal_gen/runtime/entrypoints/action/protocol.py（模块 动作接口；类别 source；类型 rename-or-move；符号 build_action_sampling_params, _build_action_model_sampling_params, _cosmos3_image_from_observation, _build_cosmos3_action_sampling_params）：Action API 协议核心：从 vla/protocol.py 重命名而来，新增 Cosmos3 特化的 action_metadata 契约与 _build_cosmos3_action_sampling_params 请求降级路径，是所有 Action API 请求进入 Cosmos3 采样的枢纽。
- python/sglang/multimodal_gen/test/unit/test_cosmos3.py（模块 单元测试；类别 test；类型 test-coverage；符号 _cosmos3_server_args, TestCosmos3ActionEndpoint, test_policy_adjusts_to_action_output, test_forward_dynamics_remains_video_output）：新增 TestCosmos3ActionEndpoint 等 6 个用例，锁定端点契约：policy/inverse_dynamics 路由、forward_dynamics 拒绝、metadata 字段、action 响应格式以及 ACTION 分支跳过 VAE 解码。
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/cosmos3.py（模块 解码阶段；类别 source；类型 data-contract；符号 Cosmos3DecodingStage.forward, Cosmos3LatentPreparationStage.forward）：Cosmos3DecodingStage 是动作输出的关键行为变更：DataType.ACTION 时直接返回动作张量，跳过 VAE decode 与 guardrails；LatentPreparation 也允许 ACTION 类型复用视觉条件。
- python/sglang/multimodal_gen/configs/sample/action.py（模块 采样参数；类别 source；类型 rename-or-move；符号 ActionSamplingParams）：VLASamplingParams 整体更名 ActionSamplingParams，并调整默认输出文件名，是动作类模型采样参数的基类变更，影响 Pi05 等所有下游。
- python/sglang/multimodal_gen/configs/sample/cosmos3.py（模块 采样参数；类别 source；类型 core-logic；符号 Cosmos3SamplingParams._adjust）：_adjust 根据 action_mode 判定是否输出 DataType.ACTION 并关闭视觉产物，是 action 输出判定的核心决策点。
- python/sglang/multimodal_gen/runtime/entrypoints/openai/video_api.py（模块 视频接口；类别 source；类型 entrypoint；符号 _build_video_sampling_params）：在 /v1/videos 入口检测 DataType.ACTION 并拒绝，保证动作请求只走 /v1/actions/generations，是 API 契约保护的关键。

- python/sglang/multimodal_gen/configs/pipeline_configs/base.py（模块 管线配置；类别 source；类型 core-logic；符号 supports_action_endpoint, supports_openpi_endpoint）：新增 supports_action_endpoint / supports_openpi_endpoint 能力位，http_server 据此路由端点挂载，是架构层面的扩展点。

关键符号：build_action_sampling_params, _build_cosmos3_action_sampling_params, _cosmos3_image_from_observation, action_metadata, Cosmos3SamplingParams._adjust, Cosmos3DecodingStage.forward, supports_action_endpoint, supports_openpi_endpoint, _build_video_sampling_params

关键源码片段

python/sglang/multimodal_gen/runtime/entrypoints/action/protocol.py

Action API 协议核心：从 vla/protocol.py 重命名而来，新增 Cosmos3 特化的 action_metadata 契约与 _build_cosmos3_action_sampling_params 请求降级路径，是所有 Action API 请求进入 Cosmos3 采样的枢纽。

```
def action_metadata(server_args: ServerArgs) -> dict[str, Any]:
    pipeline_config = server_args.pipeline_config
    # Cosmos3 是统一的 TI2V 视频管线，但 policy / inverse_dynamics
    # 模式以动作张量为输出，因此需要一份独立的 Action 契约描述
    if isinstance(pipeline_config, Cosmos3Config):
        defaults = Cosmos3SamplingParams()
        return {
            "object": "action.metadata",
            "model": server_args.model_id or server_args.model_path,
            "model_path": server_args.model_path,
            "policy_family": "cosmos3",
            "input": {
                # 支持 image 与 video 两种观测输入，分辨率来自采样参数默认值
                "modalities": ["image", "video"],
                "supported_resolutions": [
                    list(resolution) for resolution in defaults.supported_resolutions
                ],
                "state_dim": None,
            },
            "output": {
                # 动作输出为连续张量；action_dim 由具体 checkpoint 决定，
                # 这里只暴露 padded_action_dim 保证 metadata 稳定
                "action_type": "continuous",
                "action_horizon": 16,
                "action_dim": None,
                "padded_action_dim": pipeline_config.dit_config.arch_config.action_dim,
                "dtype": "float32",
            },
            "runtime": {
                "parallelism": {
                    "num_gpus": server_args.num_gpus,
```

```
                "tp_size": server_args.tp_size,
                "sp_degree": server_args.sp_degree,
                "ulysses_degree": server_args.ulysses_degree,
                "ring_degree": server_args.ring_degree,
            }
        },
        "defaults": {
            # 默认走 policy 模式，其余生成参数与配置默认值保持一致
            "action_mode": "policy",
            "action_horizon": 16,
            "num_inference_steps": defaults.num_inference_steps,
            "height": 480,
            "width": 832,
            "fps": 5,
        },
        "capabilities": {
            # Cosmos3 只暴露通用 Action API，不兼容 OpenPI websocket
            "action_modes": ["policy", "inverse_dynamics"],
            "realtime_websocket": True,
            "openpi_websocket": False,
            "batch_inputs": False,
            "multiple_candidates": False,
        },
    }

# 以下为通用动作模型的 metadata 分支（policy_family 推导），
# 字段结构与此前 vla/protocol.py 的行为保持一致
policy_family = getattr(
    pipeline_config,
    "policy_family",
    type(pipeline_config).__name__.removesuffix("PipelineConfig").lower(),
)
return {
    "object": "action.metadata",
    # ... 通用分支其余字段与原逻辑一致
}
```

python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/cosmos3.py

Cosmos3DecodingStage 是动作输出的关键行为变更：DataType.ACTION 时直接返回动作张量，跳过 VAE decode 与 guardrails；LatentPreparation 也允许 ACTION 类型复用视觉条件。

```
def forward(self, batch: Req, server_args: ServerArgs):
    """Decode latents to video, or short-circuit to raw actions for ACTION requests."""
    from sglang.multimodal_gen.runtime.pipelines_core.schedule_batch import OutputBatch

    # 首先统一提取动作预测：裁剪到真实动作维度并按需反归一化
    action_pred = None
    if getattr(batch, "action_latents", None) is not None:
```

```
    raw_action_dim = batch.extra.get("raw_action_dim")
    action_pred = batch.action_latents.float().cpu()
    if raw_action_dim is not None:
        action_pred = action_pred[:, :, :raw_action_dim]
    stats_path = getattr(batch.sampling_params, "action_stats_path", None)
    if stats_path is not None:
        method = getattr(batch.sampling_params, "action_normalization", "quantile")
        action_pred = denormalize_action(
            action_pred, method, load_action_stats(stats_path)
        )
    self.log_info(f"Action predictions shape: {tuple(action_pred.shape)}")

# 从 batch.extra 中恢复动作域信息，供 response envelope 使用
action_domain_ids = batch.extra.get("action_domain_ids")
action_domain_id = (
    int(action_domain_ids[0].item()) if action_domain_ids is not None else None
)
action_metadata = {
    "action_mode": getattr(batch.sampling_params, "action_mode", None),
    "action_domain_id": action_domain_id,
    "action_raw_action_dim": (
        batch.extra.get("raw_action_dim")
        if getattr(batch, "extra", None)
        else None
    ),
}

if batch.data_type == DataType.ACTION:
    # 动作请求直接返回动作张量，跳过视频 VAE decode、guardrails 与媒体序列化
    if action_pred is None:
        raise RuntimeError("Cosmos3 action request produced no action tensor")
    payload = {
        "request_id": batch.request_id,
        "actions": action_pred[0].numpy(),
        "action_mode": action_metadata["action_mode"],
        "domain_id": action_metadata["action_domain_id"],
        "raw_action_dim": action_metadata["action_raw_action_dim"],
        "parameters": {
            "num_inference_steps": batch.num_inference_steps,
            "num_frames": batch.num_frames,
        },
    }
    return OutputBatch(
        output=[payload],
        action_pred=action_pred,
        metrics=batch.metrics if hasattr(batch, "metrics") else None,
        **action_metadata,
    )
```

```
    # 视觉输出路径（IMAGE / VIDEO）保持原有 VAE 解码与 guardrails 流程
    is_image_gen = batch.data_type == DataType.IMAGE
    self.log_info(
        "Decoding latents to image..." if is_image_gen else "Decoding latents to video..."
    )
```

python/sglang/multimodal_gen/configs/sample/cosmos3.py

_adjust 根据 action_mode 判定是否输出 DataType.ACTION 并关闭视觉产物，是 action 输出判定的核心决策点。

```
  def _adjust(self, server_args) -> None:
      # 先判定请求是否产出动作：policy / inverse_dynamics 输出连续动作，
      # forward_dynamics 以动作为输入产出视频，仍属于视觉请求
      action_output = False
      if self.action_mode is not None:
          self.action_mode = str(self.action_mode).strip().lower()
          if self.action_mode not in (
              "policy",
              "forward_dynamics",
              "inverse_dynamics",
          ):
              raise ValueError(
                  f"Unsupported action_mode={self.action_mode!r}; expected "
                  "'policy', 'forward_dynamics', or 'inverse_dynamics'."
              )
          action_output = self.action_mode != "forward_dynamics"

      # 先执行基类的视觉路径调整（数据增强、路径展开等）
      super()._adjust(server_args)

      # 动作输出不需要任何视觉产物：关闭保存、文件路径、逐帧返回与压缩
      if action_output:
          self.data_type = DataType.ACTION
          self.save_output = False
          self.return_file_paths_only = False
          self.return_frames = False
          self.output_file_name = None
          self.output_compression = 0


  def _set_output_file_name(self) -> None:
      # 动作输出永远不需要视觉文件名，提前返回以避免对内存中的观测图做 hashing
      if self.action_mode in ("policy", "inverse_dynamics"):
          return
      # 单帧请求按 T2I 处理，派生图片扩展名；其余走视频扩展名
      if self.num_frames == 1:
          self.data_type = DataType.IMAGE
      super()._set_output_file_name()
```

评论区精华

该 PR 没有外部 review 评论，唯一一条评论是作者 mickqian 触发 CI 的 /tag-and-rerun-ci。没有可提炼的公开设计讨论；实现权衡体现在 5 个提交的演进顺序中：先落地功能，随后修复内存观测处理、跳过动作输出文件名、接受 ActionSamplingParams，最后把 vla 模块整体改名为 action 对齐模块属主。

- 暂无高价值评论线程

风险与影响

- 风险： 1) 跨模块重命名：vla 包整体改名为 action（api.py、protocol.py、openpi.py、ws_utils.py、init.py），若外部代码直接 import sglang.multimodal_gen.runtime.entrypoints.vla 会失效，仓库内部引用已全部替换但缺少兼容 alias。 2) 解码路径重构：Cosmos3DecodingStage.forward 把 action 提取逻辑前移并新增 ACTION 短路，视频 / 图像生成路径存在回归风险；单元测试用 FailIfDecoded 守护了 ACTION 分支，但没有覆盖视觉路径的端到端回归。 3) 入口契约变化：video_api 对 DataType.ACTION 抛错，若其他 pipeline 的采样参数被误标为 ACTION 会拒绝正常视频请求，目前仅 Cosmos3 会设置该类型，影响面有限。 4) 硬编码默认值：protocol.py 中 Cosmos3 分支硬编码 action_horizon=16、height=480、width=832、fps=5，若模型配置变更需同步维护。 5) CI 状态：PR Test (Extra) 显示失败（作者已触发 /tag-and-rerun-ci），最终 CI 结论需以重跑结果为准。
  - 影响：用户侧：Cosmos3 policy / inverse_dynamics 消费者获得标准 /v1/actions/generations 端点，响应为连续动作张量及 domain 元数据；继续向 /v1/videos 提交动作请求会收到 400 与提示。系统侧：动作请求在解码阶段短路，跳过 VAE decode 与 guardrails，减少不必要的视觉后处理。代码库侧：vla 模块改名 action，统一动作类模型命名；pipeline 基类新增能力位，未来新模型接入 Action API 成本降低。影响范围限于 multimodal_gen 扩散子系统，不影响 sglang/srt 核心推理路径。
  - 风险标记：跨模块重命名，入口契约变更，解码路径重构，硬编码默认值，CI 待复跑

关联脉络

- PR #34173 [diffusion] Make torch.compile opt-in for speed mode: 同属 diffusion 子系统采样参数与配置演进，且都涉及 pipeline 配置与测试的调整，体现该子系统 API 收敛方向。
- PR #34174 [diffusion] BCG: auto-capture the default warmup resolution instead of hard-requiring --warmup-resolutions: 同为 diffusion 运行时参数与部署成本简化，与本 PR 一起降低 diffusion 模型服务化门槛。
- PR #34172 [diffusion] LTX-2 quality=high fused RMSNorm+modulate + FFN GELU epilogue: 同在 multimodal_gen diffusion 管线做能力与性能扩展，说明 diffusion 子系统近期密集迭代。