

# 系统分析与设计报告（第二版）

## 1. 修订概览

项目	第一版	第二版（本次）
覆盖内容	核心实现报告	增补系统分析、需求矩阵与设计细节
文档结构	单一技术报告	分层次系统文档（背景→需求→设计→实施→测试→规划）

## 2. 项目背景与目标

Wan 视频生成管线在 HuggingFace Diffusers 中已支持多种文本/图像驱动的生成方式，但缺乏“相机控制”能力，无法复用 VideoX-Fun、CameraCtrl 等社区资产。课程项目 Phase 1 的目标是：

- 1. 提供从摄像机轨迹 txt 文件到 Plücker 射线嵌入的完整转换链路；
- 2. 兼顾 Wan 家族多条 Pipeline（T2V、I2V、VACE、Animate 等）的可复用性；
- 3. 为后续 Phase 2 的端到端相机控制扩展（例如 WanCameraControlPipeline）打下基础；
- 4. 形成可复现的技术文档与示例资料，满足课程验收与社区贡献要求。

## 3. 需求分析

### 3.1 功能性需求

编号	需求描述	实现位置	验证手段
F-01	解析 VideoX-Fun / CameraCtrl 格式 txt 轨迹	<code>process_camera_txt</code>	单元测试 + 示例脚本
F-02	支持直接传入相机参数数组	<code>process_camera_params</code>	单元测试
F-03	自动修复 frame_id 全 0 Bug	<code>fix_frame_id</code> 参数	<code>test_process_camera_txt</code>
F-04	根据目标分辨率自适应内参	<code>Camera.adjust_intrinsics</code> 逻辑	单元测试
F-05	生成 <code>[num_frames, H, W, 6]</code> Plücker 嵌入	<code>ray_condition</code>	示例脚本输出
F-06	提供 12 种内置相机轨迹样例	<code>wan_camera_samples/*.txt</code>	示例脚本
F-07	通过 CLI 快速验证	<code>wan_camera_control_example.py</code>	手动测试

### 3.2 非功能性需求

- **兼容性：** 兼容 Python 3.8+/PyTorch 1.10+，可在 CPU/GPU 上运行；
- **可维护性：** 模块化（utilities、tests、examples、docs），配套 docstring 与 README；
- **可靠性：** 每条关键路径具备 pytest 用例，示例脚本提供运行日志；
- **性能：** 81 帧、672×384 分辨率 CPU 处理 < 1s，内存 ~50MB；
- **可用性：** Quick Reference、Implementation Summary 与本报告共同提供多层次文档。

## 4. 业务场景与用例

- 1. **创作者导入现有轨迹：** 直接复用 VideoX-Fun 导出的 txt 文件；

- 2. 算法研究迭代：通过 `process_camera_params` 注入自定义轨迹，快速比较不同相机策略；
- 3. 课程实验/评审：示例脚本+测试报告可直接展示功能闭环；
- 4. 未来 Pipeline 集成：为 WanCameraControlPipeline 或 WanVACE Camera 模式提供即插即用的控制输入。

## 5. 总体架构设计

- 技术栈：Python、NumPy、PyTorch、Diffusers 基础设施、Einops；
- 模块划分：
  - *Camera Utilities* (核心算法)：位于 `src/diffusers/pipelines/wan/camera_utils.py`；
  - *Pipelines*：在 `__init__.py` 中输出公共 API，供后续各 Pipeline 引用；
  - *Tests*： `tests/pipelines/wan/test_wan_camera_utils.py` 覆盖解析、几何与错误处理；
  - *Examples & Samples*：CLI + 12 轨迹，降低上手门槛；
  - *Docs*：三层文档（快速、概要、详尽）+ 本报告。
- 数据流：txt/参数 → 解析（相机对象）→ 内参缩放 & 相对位姿 → Plücker 嵌入 → Pipeline 消费。

## 6. 关键模块详细设计

### 6.1 Camera 类

- 负责解析 19 个字段，构造 4x4 世界-相机矩阵，并保持对偶的 `w2c / c2w`；
- 提供内参缩放（宽高比适配）与位姿求逆等基础操作。

### 6.2 几何工具

- `custom_meshgrid`：兼容不同 PyTorch 版本的 `indexing` 行为；
- `get_relative_pose`：以首帧为基准生成相对位姿，满足 Wan Transformer 的时序假设；
- `ray_condition`：实现 Plücker 坐标（方向+力矩）生成，并输出 `[B, V, H, W, 6]` 张量。

### 6.3 高层 API

- `process_camera_txt`
  - I/O：txt 路径、目标分辨率、帧数裁剪/补齐、设备选择；
  - 新增参数 `fix_frame_id`，顺序化帧号；
  - 支持 `original_pose_width/height` 以处理外部轨迹的分辨率差异。
- `process_camera_params`
  - 面向高级用户，可直接传入二维数组；
  - 共享同一几何路径，保证行为一致。

### 6.4 辅助资产

- `wan_camera_control_example.py`：命令行入口，展示处理日志、张量形状与统计量；
- `wan_camera_samples/*.txt`：12 条可复现轨迹（缩放、平移、旋转、对角移动）；
- 文档矩阵：Quick Reference（速查）、Implementation Summary（概要）、Technical Report（详述）、本报告（系统视角）。

## 7. 数据与接口设计

### 7.1 Txt 轨迹格式

```
header_line
```

```
frame_id fx fy cx cy _ _ r11 r12 r13 t1 r21 r22 r23 t2 r31 r32 r33 t3
```

- 共 19 个字段，旋转矩阵需保持正交，平移单位与模型训练一致；
- `frame_id` 在第二版中可通过 `fix_frame_id` 强制递增以绕过 VideoX-Fun 缺陷。

7.2 Plücker 嵌入说明

- 每个像素射线表示为  $(\cdot, \cdot)$ ，其中方向 归一化、力矩  $=$  ；
- 输出张量排列为 `[T, H, W, 6]`，满足 Wan 模型对时空折叠的输入要求；
- 统计特征（示例 81 帧 672×384）：`[-2.4456, 2.4456]` 范围，标准差约 `0.7245`。

8. 实施与交付成果

- **代码量：** 核心模块 330 行、测试 180 行、示例 130 行、样例 txt ~6,700 行；
- **文档：** `README.md`、`QUICK_REFERENCE.md`、`IMPLEMENTATION_SUMMARY.md`、`wan-camera-control-implementation-report.md`、本报告；
- **示例资产：** 12 条轨迹涵盖缩放/平移/旋转/对角多种模式；
- **分发方式：** `pip install -e .` 形态发布，可直接被 `diffusers` 用户消费。

9. 测试与验证

- **自动化测试：** `pytest tests/pipelines/wan/test_wan_camera_utils.py -v`
  - 覆盖相机类构造、meshgrid、位姿转换、Plücker 生成、txt I/O、异常路径；
- **手动验证：** `wan_camera_control_example.py` 输出包含帧数、分辨率、统计量，便于课程答辩演示；
- **质量指标：** 关键函数均在单元测试中验证输入合法性，示例脚本可复现实验日志。

10. 运维与使用指南

1. **安装：** `cd Project/diffusers && pip install -e .`；
2. **快速体验：** 进入 `examples/community` 运行示例脚本，替换 `--camera_txt` 即可；
3. **集成：** 在任一 Wan Pipeline 中 `from diffusers.pipelines.wan import process_camera_txt`；
4. **排障：**
  - `ModuleNotFoundError` → 重新 editable 安装；
  - `ValueError: line has Y values` → 检查 txt 是否 19 列；
  - 内存不足 → 降低分辨率或帧数。

11. 风险与问题跟踪

风险/问题	描述	影响	应对策略
R-01	Pipeline 尚未接入相机嵌入	无法端到端验证	Phase 2 开发 WanCameraControlPipeline
R-02	Txt 格式单一	难以适配自定义格式	规划 JSON/关键帧插值扩展
R-03	大分辨率内存占用高	影响部署稳定性	引入分块/混合精度
R-04	社区用户学习曲线	文档分散	通过本报告与 Quick Reference 统一指引

12. 未来规划（Phase 2 展望）

1. **Pipeline 级集成：** 构建 `WanCameraControlPipeline`，在 Transformer 前向中融合 Plücker 条件；
2. **高阶特性：** 插值/平滑、轨迹可视化、批量处理、缓存与混合精度；
3. **生态互通：** 输出 ComfyUI 节点、Gradio Demo、文生视频云服务适配；

4. 研究方向：多模态控制（相机+深度+姿态）、自动轨迹生成、神经相机调度。

## 13. 参考资料与附录

- `Project/diffusers/src/diffusers/pipelines/wan/camera_utils.py`
- `Project/diffusers/tests/pipelines/wan/test_wan_camera_utils.py`
- `Project/diffusers/examples/community/wan_camera_control_example.py`
- `Project/diffusers/examples/community/wan_camera_samples/*.txt`
- `Project/diffusers/docs/source/en/api/pipelines/wan.md`
- `docs/raw/report1/wan-camera-control-implementation-report.md`