PEANUT: Prompt-Enhanced Ablation with Optical Flow-Based Neural Unit for Spatio-Temporal Consistency & VSR++ Clarity from IMG2Video Field 💫

Python v3.8 | PyTorch v2.6.0+cu124 | CUDA v12.4 | TorchAudio v2.6.0+cu124 | TorchVision v0.21.0+cu124 | imageio-ffmpeg v0.6.0

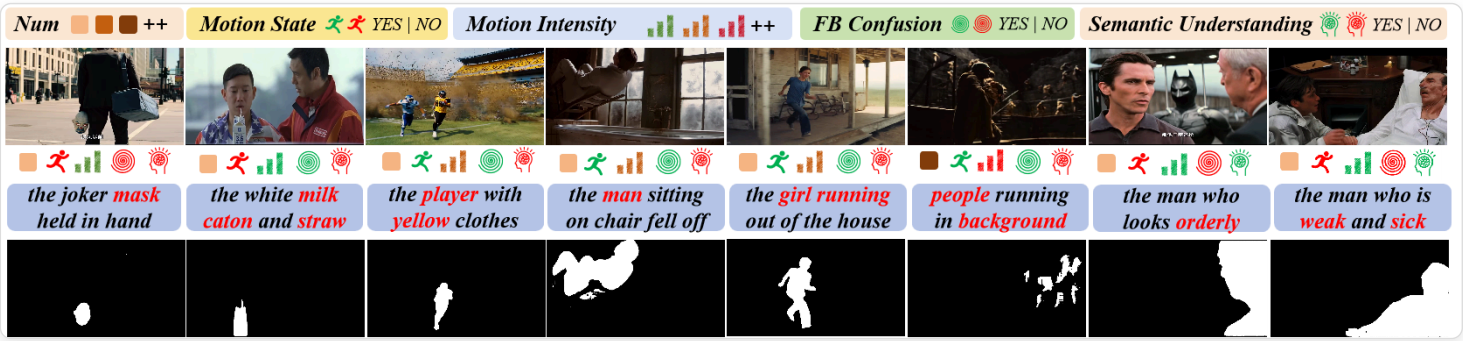
Contact us here and welcome your coming !

Name: Minghao Li (Aiur)

Affiliation: Beijing Normal University - Hong Kong Baptist University (BNBU)

Email: t330034027@mail.uic.edu.cn

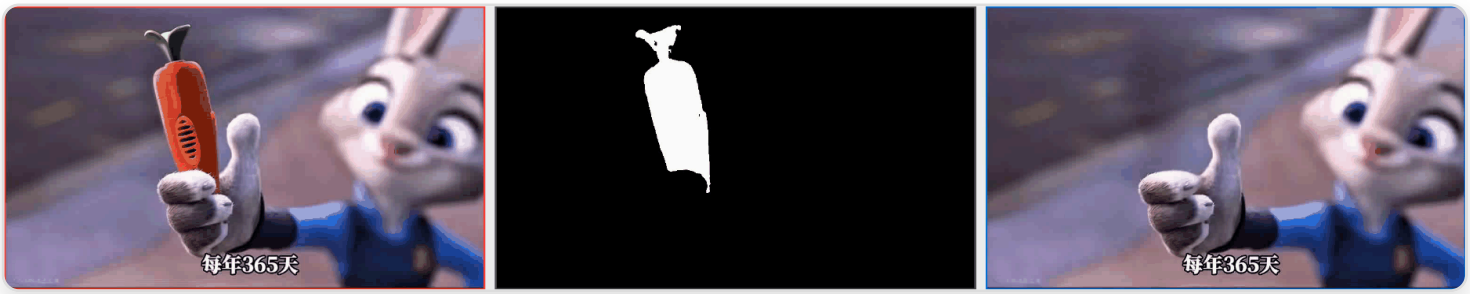
P-MASK MODULE IN DIFFERENT CHARACTERISTIC SCENES



🎬 💫 Let's Enjoy the Show

🤗 Initial Video → 🐑 Prompt-Guided Mask → 🥳 Inpaint Video

🟨 Rapid and Continuous Movement 🟩 Spanning Large Area 🟥 Slow Motion 🤗 Prompt: flying chains apart from its tips 🔍 Check out Alita:Fight Angle 🤹



🟨 Semantic Understand 🟩 Screen Jump 🟥 Background and Object Classify 🤗 the carrot 🟰 🤗 the thing held by rabbit 🟰 🤗 the thing clicked at 1st shot 🔍 Check out your Officer Judy !

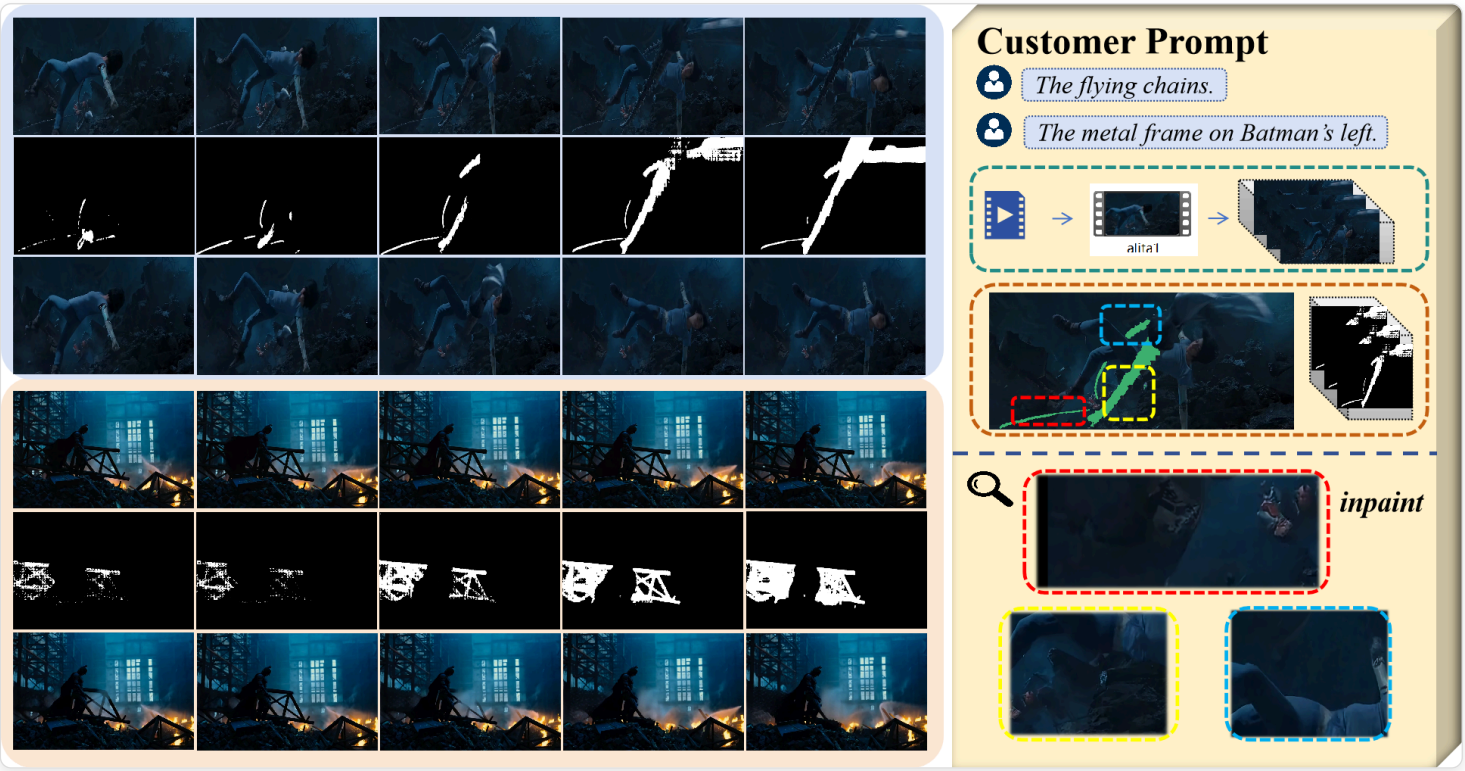
🟨 Semantic Understand 🟩 Slow Motion 🟥 Background and Object Classify 🤗 Prompt: the person setting on chair fell off 🔍 Check out to Time Reversal !

🟨 Semantic Understand 🟩 Long-duration Movement 🟥 Complex Background 🤗 the person running out with yellow clothes 🟰 🤗 the person who runs faster 🔍 Check out to find whether he also fell off ?

🤗 Prompt: the girl who runs out of the house 🔍 Check out Murphy, click !

> *🤗 Frame Processing and Detail Capture of Action Shots

🎬 PIPELINE VISUALIZATION



📁 1. Folder Structure

```
SuperVideo-inpaint/
├── raw_video/              # patch processing raw video input
├── results/                # patch processing inpaint result (IMG/GIF)
├── demo_video/             # single processing demo video input
├── demo_output/            # single processing demo video output
│
├── frames_package/         # temp: video2frames (fps/extract-rate settings)
├── mask_package/           # temp: mask generation based on frame_package with P-MASk processed
├── inpaint_package/        # temp: NOF-Eraser output storage
├── restore_package/        # temp: UR-Net output storage
├── denoise_package/        # temp: nlm denoise processed
│
├── experiment/             # exp & eval metrics
│   └── exp0/               # the fail of Big-LaMa implemented in video inpaint
│   └── exp1/               # P-MASk Hyperparameters: model-size / clip-window / step
│   └── exp2/               # NOF-Eraser Hyperparameters: neighbors / step

│
├── SAMWISE/                # project 1 P-MASk: prompt guied mask generation for each sampled frames
├── E2FGVI_Project/         # project 2 NOF-Eraser: flow-based video inpaint
├── BasicVSR_PlusPlus/      # project 3 UR-Net & Denoise Block for video clearity
│
├── video_inpaint_pipeline_en.ps1  # ⭐💫 main function to patch process
└── video_inpaint_demo.ps1         # ⭐ demo single process
```

🚀 2. Quick Start (Demo 🎦 ➕ Patch 🎭)

2.1 Method 1: Demo Script 🎦

Single Video for the whoe processing pipeline.

```
# video to ur demo_video package
Copy-Item "your_video.mp4" -Destination "demo_video\"
# run
.\video_inpaint_demo.ps1 -VideoPath "your_video.mp4" -TextPrompt "the object to remove"
# model saved in: demo_output\your_video_result\
```

Advantage:

- ✓ Automatically creates a separate output folder
- ✓ Retains all intermediate results (frames, masks, repairs, enhancements)
- ✓ Suitable for single video testing and debugging
- ✓ Supports custom resolution, FPS, and other parameters

2.2 Method 2: Batch Processing Pipeline 🎭

batch processing in `raw_video` package video

- basic pipeline version (all)

```
# put video into ur raw_video package
Copy-Item "video.mp4" -Destination "raw_video\"
# bathc processing pipeline
.\video_inpaint_pipeline_en.ps1 -VideoPath "raw_video\video.mp4" -TextPrompt "the object"
# results saved in results package
```

- advanced pipeline version

```
# skip certain steps
.\video_inpaint_pipeline_en.ps1 `
    -VideoPath "raw_video\video.mp4" `
    -TextPrompt "object" `
    -SkipMask `
    -SkipInpaint
```

Advantage:

- ✓ Supports specifying the output directory (-OutputDir)
- ✓ Automatically cleans up intermediate files
- ✓ Suitable for batch processing of multiple videos

📝 3. Use Demo

3.1 Demo 1: Demo Basic Usage

```
.\video_inpaint_demo.ps1 -VideoPath "alita1.mp4" -TextPrompt "the flying chains and its lips"
```

3.2 Demo 2: Custom Resolution and FPS

- max resolution (default 1080)
- fps (default 30)

```
.\video_inpaint_demo.ps1 `
    -VideoPath "batman1.mp4" `
    -TextPrompt "the person walking on the left" `
    -MaxResolution 1080 `
    -FrameExtractionFps 30
```

3.3 Demo 3: Fast Mode (CPU mode, Skip enhancements ❎)

```
.\video_inpaint_demo.ps1 `
    -VideoPath "test.mp4" `
    -TextPrompt "the watermark" `
    -SkipEnhance `
    -ForceCPU
```

3.4 Demo 4: High Quality Mode (Slower GPU mode, All pipeline ✅)

- maximum settings with All Settings Pipeline

```
.\video_inpaint_demo.ps1 `
    -VideoPath "video.mp4" `
    -TextPrompt "the logo" `
    -NeighborStride 2 `
    -MaxLoadFrames 12 `
    -DenoiseStrength 7
```

⚙️ 4. Parameter Description

4.1 Demo Script (video_inpaint_demo_en.ps1)

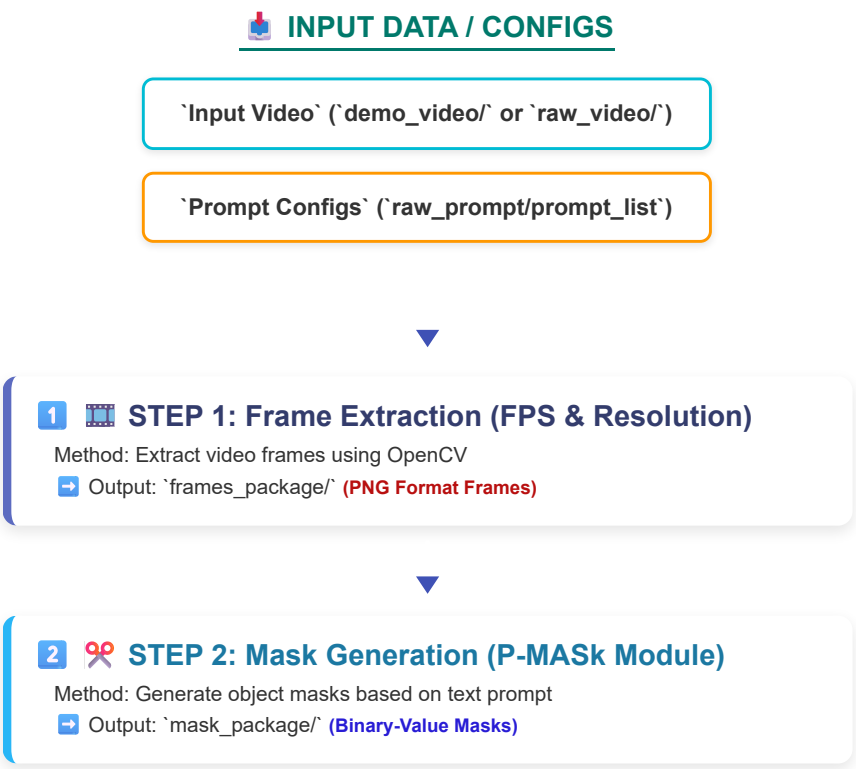
Parameters	Necessity	Default	Statement
`-VideoPath`	✓	-	demo_video the path od video
`-TextPrompt`	✓	-	the removed object in natural language
`-MaxResolution`	✗	720	max resolution，limit frames' height /width
`-FrameExtractionFps`	✗	30	Frame Extractions FPS
`-NeighborStride`	✗	3	Neighborhood step size (1-10, smaller values result in better quality but slower speed)
`-MaxLoadFrames`	✗	8	Maximum number of frames loaded (affects memory usage and quality)
`-DenoiseStrength`	✗	5	Noise reduction intensity (1-10)
`-SkipMask`	✗	✗	Skip P-MASk Processing
`-SkipInpaint`	✗	✗	Skip NOF-Eraser Video Inpainting
`-SkipEnhance`	✗	✗	Skip UR-Net Video Enhancement
`-SkipDenoise`	✗	✗	Skip Denoise

Parameters	Necessity	Default	Statement
-ForceCPU	✗	✗	Force CPU（BasicVSR++）

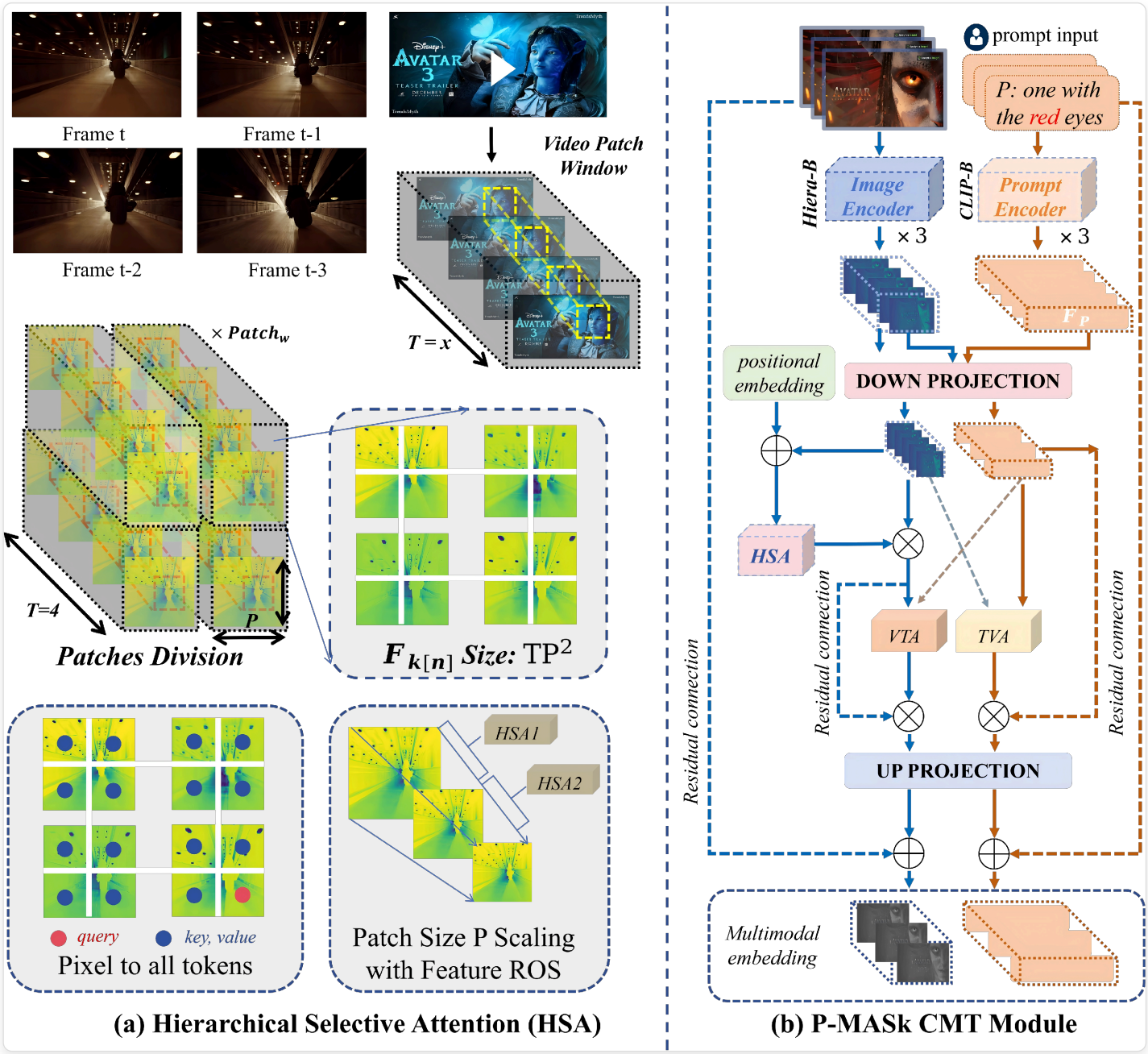
4.2 Batch Processing (video_inpaint_pipeline_en.ps1)

Parameters	Necessity	Default	Statement
-VideoPath	✓	-	Video file path (supports relative and absolute paths)
-TextPrompt	✓	-	Description of the object to be removed (English)
-OutputDir	✗	"results"	Output directory (relative or absolute path)
-NeighborStride	✗	3	Neighbor stride
-MaxLoadFrames	✗	8	Maximum number of frames to load
-OtherParams	✗	-	Similar to the demonstration script

🚀 5. PEANUT Video Processing Architecture



P-MASK MODULE PIPELINE PROCESSING



▼

3 ⚡ **STEP 3: Resizing (Frame & Mask Speedup 💫)**

Method: Adjust to specified resolution

➡️ Output: `*_resized` folder

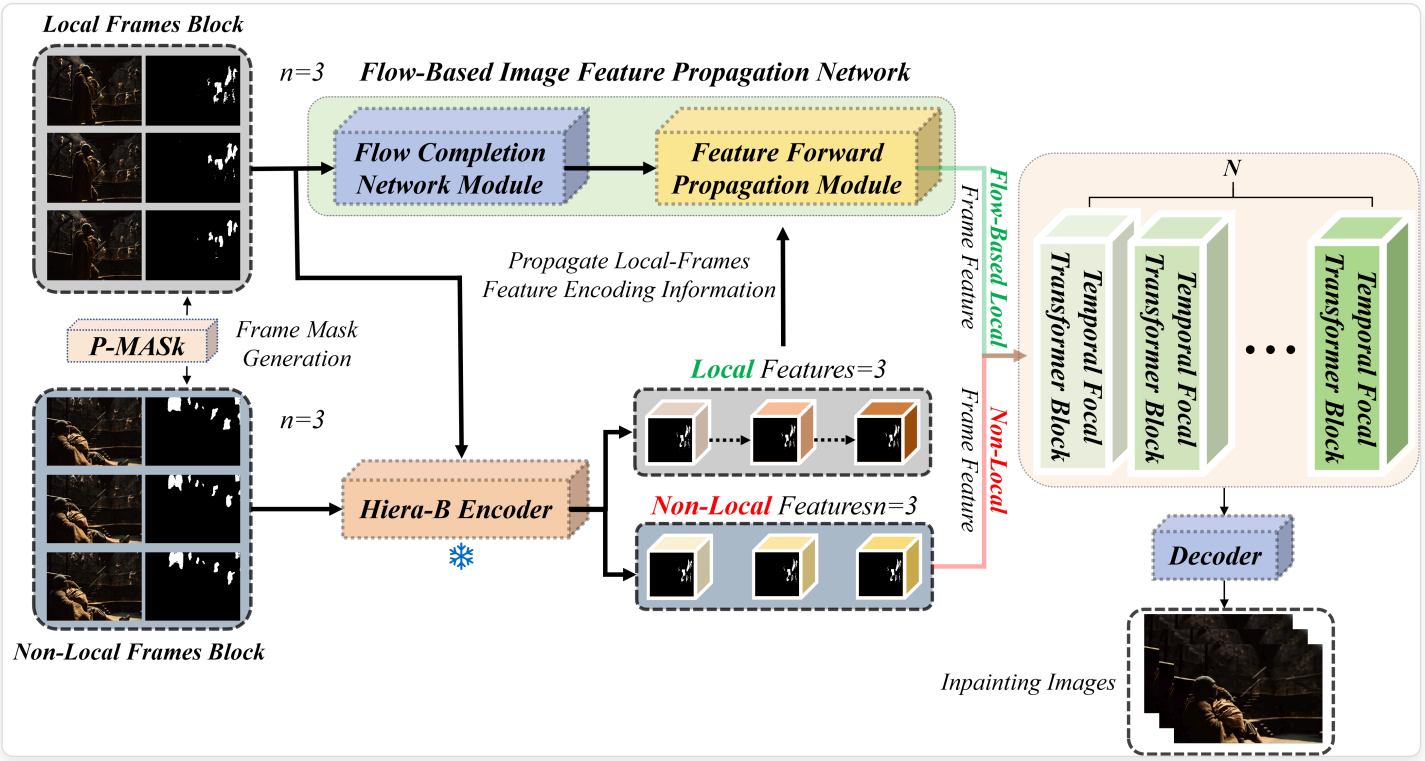
▼

4 🪄 **STEP 4: Video Inpainting (NOF-Eraser CORE)**

Method: Remove masked objects (E2FGVI-based)

➡️ Output: `inpaint_package/` **(Inpainted Frames)**

🎬 NOF-ERASER MODULE PIPELINE PROCESSING



▼

5 🌟 STEP 5: Quality Enhancement (UR-Net with BasicVSR++ Algorithm)

Method: Improve video quality and clarity

➡️ Output: `restore_package/` **(Enhanced Frames)**

▼

6 🧼 STEP 6: Denoising Module (NLM Denoise)

Method: Apply NLM/Bilateral/Gaussian denoising

➡️ Output: `denoise_package/` **(Denoised Frames)**

▼

7 🎥 STEP 7: Video Generation

Method: Create final video from frame sequence

▼

✅ **FINAL OUTPUT**
`demo_output/` or `results/`

💡 6. Performance Recommendations

6.1 Quick Mode (Low Quality & Fast 🏃)

```
-NeighborStride 5 -MaxLoadFrames 4 -DenoiseStrength 3
```

6.2 Balanced Mode (Recommended 🥳)

```
-NeighborStride 3 -MaxLoadFrames 8 -DenoiseStrength 5
```

6.3 Quality Mode (High Quality & Slow Speed 🙇)

```
-NeighborStride 2 -MaxLoadFrames 12 -DenoiseStrength 7
```

⚠️ 7. Important Operating Notes

> Please follow these guidelines before running the processing script to ensure optimal performance and results.

👉 1. Video Path Restriction

You only need to provide the filename (e.g., `alita1.mp4`). The video file must be placed inside the `raw_video/` folder.

🗣️ 2. Text Prompt Language

The object to be removed must be described using English text prompts (e.g., "the person", "the car", "the traffic sign").

💾 3. Memory Usage (VRAM)

If you encounter Out-of-Memory (OOM) errors, please try reducing the value of the `MaxLoadFrames` parameter.

⏱️ 4. Processing Time Estimate

Processing time depends on video length and parameter settings. A 10-second video typically requires 5 to 20 minutes for full pipeline execution.

🧹 5. Clean Run Policy

Each new script run will clear all previous intermediate results (e.g., frames, masks) to maintain a clean directory structure.

🛠️ 8. Environment Requirements

🖥️ 8.1 Operating System

> **Windows | CUDA: 12.7 | PyTorch Version: 2.6.0+cu124 CUDA Available: True CUDA Version: 12.4**

🧪 8.2 Conda Environments (Dependencies)

The processing pipeline requires three dedicated Conda environments for different components:

Conda Environment: `samwise`
→ **Function:** Mask Generation (SAMWISE Project)

Conda Environment: `e2fgvi-project`
→ **Function:** Video Inpainting (E2FGVI Project)

Conda Environment: `basicvsrpp-demo`
→ **Function:** Quality Enhancement (BasicVSR++ Project)

🎬 9. Video Content Inpainting Workflow Examples

The following provides two structured workflows using PowerShell for a hypothetical video inpainting project (assumed to be `SuperVideo-inpaint`).

I. Demonstration Script Workflow: Rapid Single-Video Testing ✨

This workflow is optimized for quickly verifying the project setup, the performance of the **segmentation model** (Mask Generation), and the final **inpainting quality** on a single test video.

Step	Command (PowerShell)	Purpose & Technical Explanation
1. Navigate to Project Root Path	`cd C:\Users\Aiur\SuperVideo-inpaint`	Change directory to the repository's root, ensuring correct path resolution for all dependencies and scripts.
2. Data Ingestion	`Copy-Item "D:\alita1.mp4" -Destination "demo_video\"`	Ingest the source video into the designated input folder (`demo_video/`) for the demonstration environment.
3. Core Execution	`.\video_inpaint_demo.ps1 -VideoPath "alita1.mp4" -TextPrompt "the flying chains and its lips"`	Executes the primary pipeline: 1. **Frame**

Step	Command (PowerShell)	Purpose & Technical Explanation
		Extraction. 2. **Prompt-based Segmentation/Mask Generation** (e.g., using SAM or RITM). 3. **Video Inpainting** (e.g., using a Flow-based or RNN model).
4. Final Output Review	`explorer demo_output\alita1_result`	Quickly opens the directory containing the processed, high-quality output video (`alita1_result.mp4`).

🔬 Intermediate Results Analysis (Debugging Phase)

For professional debugging and **Quantitative Evaluation** of the model's performance (e.g., checking Mask IoU or Inpainting Consistency), these folders are crucial:

- `explorer frames_package\alita1` : Review the original **frame sequence**.
- `explorer mask_package\alita1_mask` : Examine the generated **temporal mask sequence**, which defines the target region for removal.
- `explorer inpaint_package\alita1` : Analyze the per-frame **inpainting results** before final video composition/post-processing.

II. Batch Pipeline Workflow: Large-Scale Processing and Data Engineering ✨

This workflow is designed for running multiple jobs, facilitating large-scale **data science experiments** and providing control over output organization.

Step	Command (PowerShell)	Purpose & Technical Explanation
1. Bulk Data Preparation	`Copy-Item "D:\videos\*.mp4" -Destination "raw_video\"`	Standardizes input by copying a batch of raw videos into the designated processing queue folder (`raw_video/`).
2. Sequential Processing (Job 1)	`.\video_inpaint_pipeline_en.ps1 -VideoPath "raw_video\video1.mp4" -TextPrompt "the person"`	Initiates the pipeline to remove the target object ("the person") from `video1.mp4` .

Step	Command (PowerShell)	Purpose & Technical Explanation
3. Sequential Processing (Job 2)	`.\video_inpaint_pipeline_en.ps1 -VideoPath "raw_video\video2.mp4" -TextPrompt "the car"`	Initiates the pipeline to remove the target object ("the car") from `video2.mp4`.

⚙️ Custom Output Directory for Experiment Tracking

The `-OutputDir` parameter is vital for **Mathematical Modeling/Experimentation**, allowing results from different models or parameter configurations to be isolated for comparison and analysis.

```
.\video_inpaint_pipeline_en.ps1 `
    -VideoPath "raw_video\video3.mp4" `
    -TextPrompt "the logo" `
    -OutputDir "custom_output\LogoRemoval_ExperimentA"
```

📊 10. Quality Assessment & Metric Evaluation Workflow

This section outlines the workflow for running **EXP0**, a controlled experiment designed to rigorously evaluate the inpainting quality of the **LAMA4Video** model against established metrics.

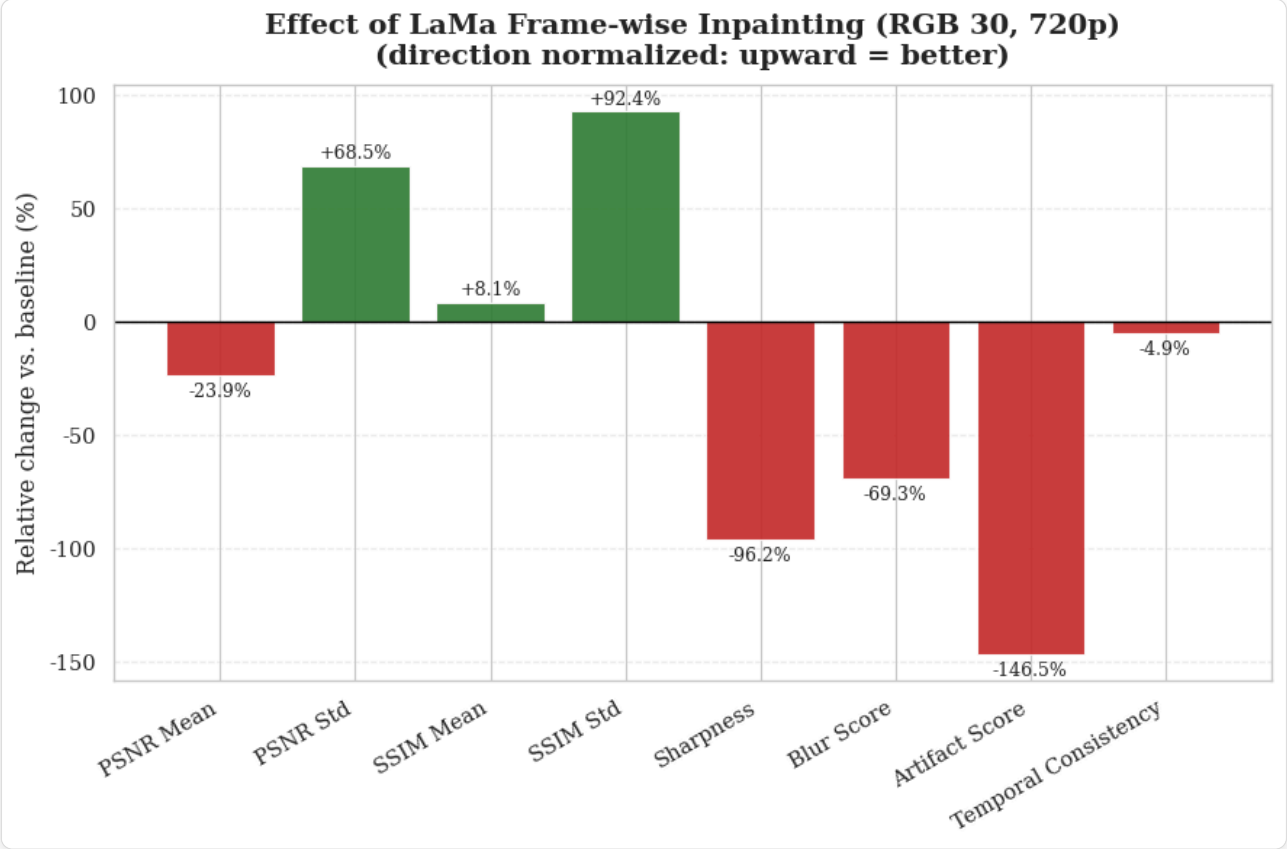
Step	Command (PowerShell)	Purpose & Technical Explanation
1. Change Directory	`cd experiment\exp0`	Navigate to the specific experiment directory to ensure the evaluation scripts can locate the necessary ground truth data and model outputs.
2. Environment Activation	`conda activate e2fgvi-project`	**Crucial Step:** Activate the designated Anaconda/Conda environment. This environment must contain all dependencies (e.g., PyTorch, NumPy, scikit-image) required by the `e2fgvi-project` for complex metric calculations.
3. Execute Evaluation	`python evaluate_exp0_quality.py`	Run the script to compute raw quantitative metrics by comparing the LAMA4Video output with the ground truth (GT) frames. This generates the primary data for analysis.
4. Generate Visual Reports	`python visualize_exp0_quality.py`	Creates detailed, per-video visualizations (e.g., charts, plots) for metric trends over time or frame sequence.
5. Generate Overall Summary	`python visualize_overall_exp0_metrics.py`	Generates aggregated reports (e.g., bar charts, box plots) summarizing the **mean** and **variance** of metrics across the entire dataset.
6. Review Results	`explorer results\metrics`	Opens the output directory containing the generated CSV/JSON metric files and the final visualization plots (e.g., PNG, SVG).

📈 Core Evaluation Metrics

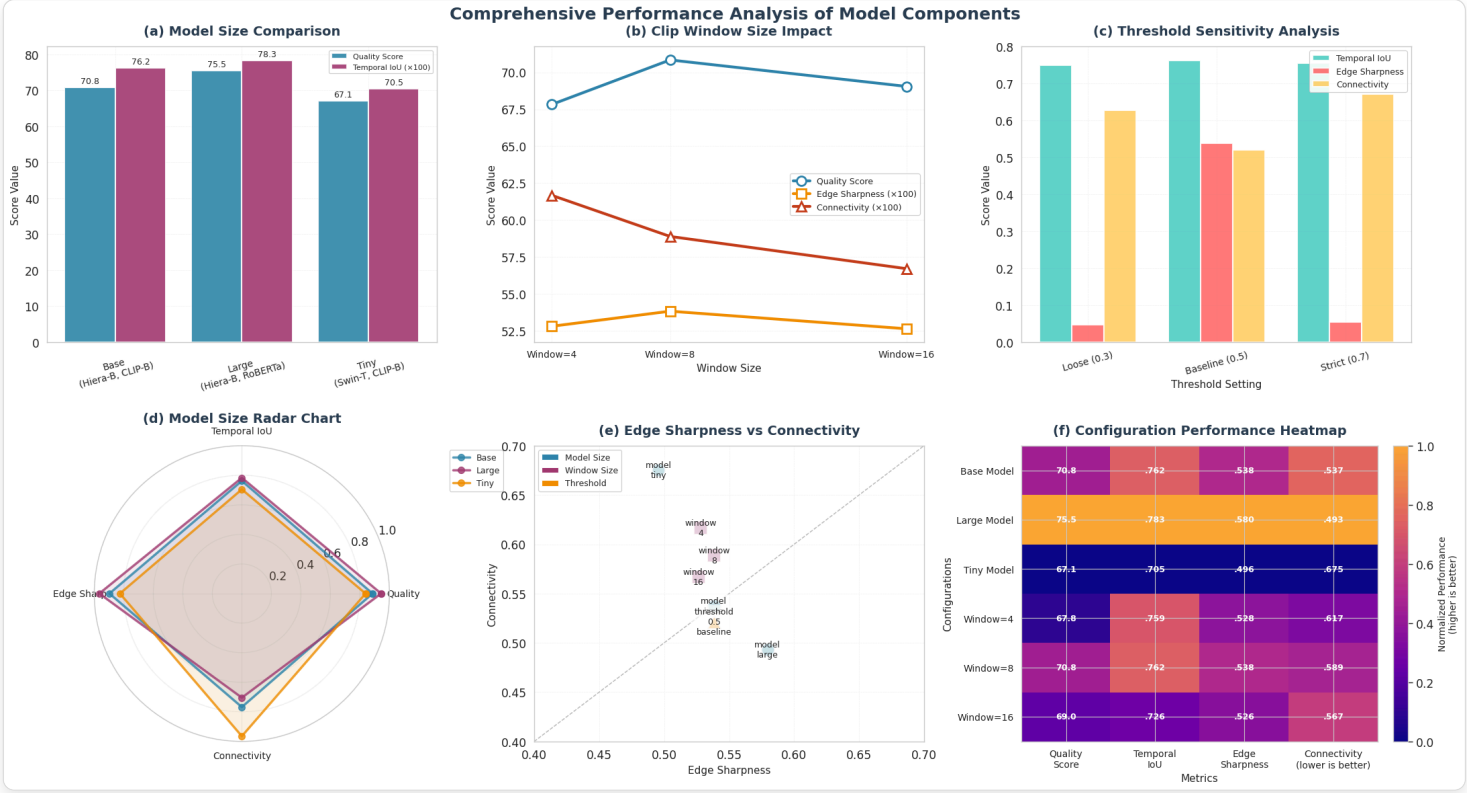
The experiment focuses on both traditional signal processing metrics and specialized metrics for video consistency:

Metric Name (Full)	Acronym	Domain & Relevance
Peak Signal-to-Noise Ratio	PSNR	Measures absolute reconstruction fidelity based on Mean Squared Error (MSE). Higher is better.
Structural Similarity Index Measure	SSIM	Assesses perceptual similarity based on luminance, contrast, and structure, aligning more closely with human vision.
Sharpness Score	-	Measures the clarity of edges and fine details in the output frames (e.g., using Laplace filter variance).
Blur Score	-	Quantifies the inverse of sharpness, indicating lack of detail and over-smoothing. Lower is better.
Artifact Score	-	Specialized metric to detect common inpainting defects like checkerboard patterns or color bleeding. Lower is better.
Temporal Consistency	-	**Critical for Video Inpainting.** Evaluates frame-to-frame smoothness and stability of the repaired region, often via optical flow or difference mapping.

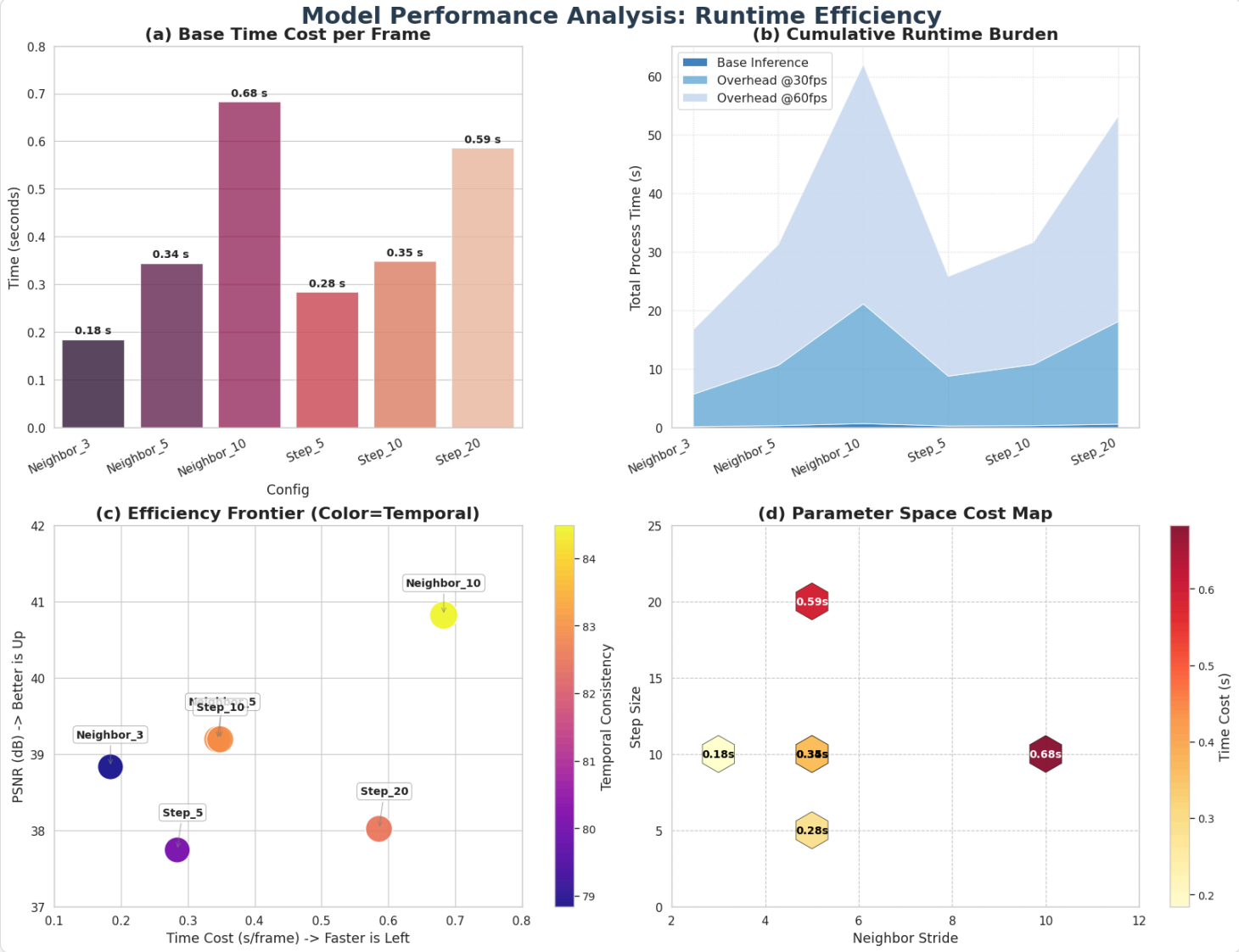
🎬 EXPERIMENT0: THE FAIL OF BIG-LAMA APPLIED IN VIPAINT



🎬 EXPERIMENT2: P-MASK HYPERPARAMETERS COMPARISION



🎬 EXPERIMENT2: NOF-ERASER HYPERPARAMETERS COMPARISION



11. Troubleshooting Guide

❌ Issue 1: Video File Not Found

💡 **Solution:**

- Demo Script: Ensure the video is placed in the `demo_video/` folder.
- Batch Script: Use the full path or correct relative path (e.g., `raw_video\video.mp4`).

Issue 2: ModuleNotFoundError (cv2, PIL, etc.)

💡 **Solution:**

The script is designed to automatically activate the correct Conda environment. Ensure all three required environments are properly installed:

```
conda env list  # Check your environments
```

💾 Issue 3: CUDA Out of Memory (OOM)

💡 Solution:

Reduce VRAM consumption by decreasing these parameters:

```
-NeighborStride 5 -MaxLoadFrames 4
```

🚫 Issue 4: Mask Count Mismatch (IndexError)

Symptom:

```
IndexError: index X is out of bounds
```

💡 Solution: Re-generate Masks

Clear old intermediate data before re-running:

```
# Clear old data
Remove-Item "frames_package\video*" -Recurse -Force
Remove-Item "mask_package\video*" -Recurse -Force
.\video_inpaint_demo.ps1 -VideoPath "video.mp4" -TextPrompt "object"
```

✅ Issue 5: SAMWISE Output Path Error

Symptom:

```
Binary masks not found
```

💡 Solution:

This issue includes an automatic fix in the script. Ensure you are using the latest version of `video_inpaint_demo.ps1`.

🔄 12. Version History

🟢 v3.0 (2025-12-5)

- ✨ Feature: Added the `video_inpaint_demo.ps1` demonstration script.
- ✨ Feature: Introduced the EXP0 Quality Assessment System.
- 🐛 Fix: Resolved automatic Conda environment activation issues.
- 🐛 Fix: Fixed SAMWISE path detection problems.
- 🐛 Fix: Fixed frame/mask count validation logic.
- ⚙️ Update: `video_inpaint_pipeline_en.ps1` now includes the `-OutputDir` parameter.

🟡 v2.0 (2025-11-14)

Initial Release.

Current Version: 3.0