Warlock-Studio

– User Manual & Technical Documentation –

Version 4.0

Iván Eduardo Chavez Ayub
@Ivan-Ayub97 on GitHub

July 20, 2025

Abstract

Warlock-Studio 4.0 Technical Documentation

This comprehensive technical manual serves as the definitive guide for Warlock-Studio 4.0 "SuperResolution Revolution". The documentation encompasses detailed architectural analysis, implementation specifics, optimization strategies, and comprehensive troubleshooting procedures. Information has been validated through extensive source code analysis and real-world testing scenarios.

Target Audience: Technical users, system administrators, content creators, and AI enthusiasts seeking in-depth understanding of AI-powered media enhancement workflows.

Coverage: Installation procedures, system architecture, AI model specifications, performance optimization, error resolution, and advanced use cases.

Contents

1. Introduction

Warlock-Studio is an AI-powered media enhancement and upscaling suite, designed to deliver high-quality results through an accessible user interface. Version 4.0 introduces a revolutionary smart AI model distribution system that automatically downloads required components on first launch, reducing installer size from 1.4GB to just 150MB. Additionally, it includes key enhancements in GPU management, error handling, and performance efficiency, solidifying its position as a powerful tool for professional content creators.

1.1. What's New in Version 4.0 - "SuperResolution Revolution"

The latest version brings the following key enhancements:

- **Smart AI Model Distribution:** Revolutionary lightweight installer (150MB vs 1.4GB) with automatic AI model download system that fetches models (327MB) on first launch.
- **SuperResolution-10 Model Integration:** Revolutionary 10x upscaling capabilities specifically designed for very low-resolution images, perfect for bringing old photos back to life.
- **Enhanced AI Architecture:** Implemented robust 'AI_model_base' class with comprehensive error handling and GPU acceleration support.
- **Code Quality Improvements:** Fixed critical import errors, consolidated duplicate code sections, and improved type annotations for better maintainability.
- **Improved Error Handling:** Added graceful degradation mechanisms that prevent crashes and provide meaningful error messages.
- **Complete Model Integration:** SuperResolution-10 is fully integrated into UI, processing pipeline, and information dialogs with proper VRAM management.

1.2. Main Features

- **AI Upscaling:** Uses state-of-the-art models like Real-ESRGAN, BSRGAN, and SRVGGNet-Compact.
- **Frame Interpolation:** Increases FPS or creates smooth slow-motion effects using RIFE models.
- **Noise Reduction:** Includes dedicated IRCNN models for cleaning images and videos.
- **Face Restoration:** GFPGAN model for enhancing and restoring faces in photos.
- **Hardware Acceleration:** Uses the ONNX Runtime engine with the DirectML provider (`DmlExecutionProvider`) for GPU acceleration compatible with DirectX 12.
- **Advanced Video Encoding:** Supports hardware-accelerated encoders from NVIDIA (NVENC), AMD (AMF), and Intel (QSV).
- **Smart Model Distribution:** Automatic AI model download system (327MB) on first launch, reducing installer size from 1.4GB to 150MB.

2. Installation and Program Architecture

2.1. Smart AI Model Distribution System

Warlock-Studio 4.0 introduces a revolutionary AI model distribution system that automatically downloads required components during the application's first launch.

2.1.1 Advantages of the New System

- **Lightweight Installer:** 78% size reduction (from 1.4GB to 150MB)
- **Smart Download:** AI models (327MB) are automatically downloaded on first use
- **Multiple Sources:** Redundant URLs (GitHub, SourceForge) ensure availability
- **Progress Tracking:** Visual indicators show download speed and completion percentage
- **Error Recovery:** Automatic retry mechanisms and clear error messages
- **Resume Support:** Interrupted downloads can be resumed seamlessly

2.1.2 Automatic Download Process

When starting Warlock-Studio for the first time, the system will perform the following operations:

1. **Model Verification:** Checks if AI models are present
2. **Confirmation Dialog:** Requests permission to download 327MB of AI models
3. **Progressive Download:** Shows real-time progress with visual indicators
4. **Integrity Validation:** Verifies that downloaded files are complete
5. **Automatic Extraction:** Decompresses and organizes models in the correct structure

2.1.3 Offline Setup and Manual Configuration

For users with limited connectivity or specific preferences:

Offline Installation:
Models can be downloaded separately and manually placed in the `AI-onnx` folder

Manual Location:
Download `AI-onnx-models.zip` from GitHub Releases and extract to the installation directory

File Verification:
The application will automatically verify the presence of all required models

2.2. System Requirements

Component	Requirement
Operating System	Windows 10 (64-bit) or later
RAM	8 GB (Minimum), 16 GB (Recommended)
Graphics Card (GPU)	**DirectX 12** compatible. **Recommended: 4+ GB VRAM**.
Storage	2 GB of free space. An SSD is recommended for better performance.

Table 1: Hardware and software requirements for Warlock-Studio 4.0.

2.3. File Structure and Dependencies

Warlock-Studio is a self-contained application. The following components are included in the installation and require no action from the user.

- `ffmpeg.exe:` Located in the `Assets` folder, it is the engine for all video manipulation, encoding, and decoding.
- `exiftool.exe:` Also in `Assets`, it is used to read and write metadata (EXIF, XMP), ensuring that the original file information is preserved.
- **AI Models:** The models in `.onnx` format are located in the `AI-onnx` folder.
- **User Preferences:** A file named `Warlock-Studio_4.0_UserPreference.json` is saved in the user's **Documents** folder.
- **Logs:** Log files are stored in `Documents\Warlock-Studio_4.0_Logs`.

3. Detailed Guide to AI Models

The choice of AI model is the most important factor for quality and processing time.

3.1. Model Comparison Table

The following table details the relative VRAM usage of each model.

Model	Function	Scale	VRAM (GB)	Recommended Use Case & Details
			Denoising Models	
`IRCNN_Mx1`	Denoise	x1	4.0	Moderate noise reduction. Good for cleaning old photos with medium artifact levels.
`IRCNN_Lx1`	Denoise	x1	4.0	Intense noise reduction. Best for heavily degraded images with severe artifacts.
			High-Quality Upscaling Models (Slower Processing)	
`BSRGANx4`	Upscale	x4	0.6	Realistic photos with excellent fine detail preservation. Best for portraits and natural scenes.

Model	Function	Scale	VRAM (GB)	Recommended Use Case & Details
`BSRGANx2`	Upscale	x2	0.7	Similar quality to x4 variant but for moderate upscaling needs. Faster processing.
`RealESRGANx4`	Upscale	x4	0.6	General purpose model. Excellent for textures and mixed content types.
`RealESRNetx4`	Upscale	x4	2.2	Alternative to RealESRGAN. May offer better speed-quality balance on some systems.
High-Speed Upscaling Models (Lightweight)				
`RealESR_Gx4`	Upscale	x4	2.2	Fast processing ideal for videos. Good balance of speed and quality.
`RealESR_Animex4`	Upscale	x4	2.2	Specialized for anime, cartoons, and illustrated content. Preserves artistic style.
Face Restoration Models				
`GFPGAN`	Restore	x1	1.8	AI-powered face enhancement and restoration. Repairs damaged faces in old photos.
Frame Interpolation Models (Video Only)				
`RIFE`	Interpolate	N/A	N/A	Maximum interpolation quality. Creates smooth motion between frames for high FPS.
`RIFE_Lite`	Interpolate	N/A	N/A	Optimized version for GPUs with limited VRAM ($<$ 4 GB). Faster processing.
Super Resolution Models				
`SuperResolution-10`	Upscale	x10	0.8	Extreme upscaling for very low-resolution images. Perfect for restoring old, small photos.

Table 2: Comprehensive guide to AI model selection and VRAM requirements.

4. Configuration and Performance Optimization

4.1. Critical Performance Parameters

- **Input Resolution %:** The most effective adjustment for speed. It reduces the resolution before processing it with AI. A value between **50% and 75%** is usually ideal.
- **GPU VRAM Limiter (GB):** Define your GPU's VRAM. It is used to calculate the size of the processing *tiles* and prevent memory errors.
- **AI Multithreading:** For videos only. It processes multiple frames in parallel, speeding up the process but consuming more VRAM and CPU.
- **AI Blending:** Blends the original image with the processed image. Useful for reducing artifacts when using a low *Input Resolution*.
- **Frame Generation:** For RIFE models, allows creating interpolated frames for higher FPS or slow-motion effects.

4.2. The User Preferences File

The Warlock-Studio4.0 UserPreference.json file saves your settings.

JSON Key	Description
`default_AI_model`	The last selected AI model.
`default_AI_multithreading`	The number of processing threads for video.
`default_gpu`	The last selected GPU (Auto, GPU 1, etc.).
`default_keep_frames`	Whether to keep the video frames ("ON" or "OFF").
`default_image_extension`	Default image extension (`.png`, `.jpg`, etc.).
`default_video_extension`	Default video extension (`.mp4`, `.mkv`, etc.).
`default_video_codec`	The default video encoder (x264, hevc_nvenc, etc.).
`default_blending`	The selected blending level (Low, Medium, High).
`default_output_path`	The last selected output path.
`default_input_resize_factor`	The input resolution percentage value.
`default_output_resize_factor`	The output resolution percentage value.
`default_VRAM_limiter`	The GPU VRAM limiter value.

Table 3: Keys saved in the user preferences file.

5. Advanced Troubleshooting Guide

Warning

The **Number 1** cause of errors is **special characters** in file paths and names. Avoid using: `'`, `"`, `@`, `#`, `$`, `%`, `&`, `*`, `[`, `]`, `?`, `etc.`.

Error: "FFmpeg encoding failed: Invalid argument"
Cause: Invalid file name or path. **Solution:** Rename the file and/or its containing folder, removing any special characters.

Error: "out of memory" or unexpected crash
Cause: The GPU ran out of video memory (VRAM). **Solution:**

1. Lower the **VRAM Limiter** to a value equal to or less than your GPU's actual VRAM.
2. Lower the **Input Resolution %** to 75% or less.
3. For videos, decrease the **AI Multithreading** threads or turn it "OFF".
4. The application will try to recover from this error automatically.

Error: "cannot convert float NaN to integer"
Cause: GPU driver timeout, often due to overload or overheating. **Solution:** Restart the process **without deleting the generated frames folder**. The application will read the existing frames and resume work from where it failed.

Issue: Output video has no audio
Cause: The original video had no audio track, a *Slowmotion* mode was used, or the audio codec was incompatible. **Solution:** The program first tries to copy the audio stream directly. If that fails, it tries to re-encode to AAC. If all fails, it saves the video without audio. Using the `.mkv` container for the output may help.

Issue: Application won't open or closes on startup
Cause: Corrupt settings, lack of permissions, or an environment error. **Solution:**

1. Go to your **Documents** folder and delete the

⚠2**Issue: Frame interpolation not working**
Cause: RIFE models are not selected or incompatible video format. **Solution:** Ensure you have selected a RIFE model (RIFE or RIFE_Lite) and that the frame generation option is properly configured.

6. Advanced Architecture and Processes

6.1. Inference Engine and Hardware Acceleration

Warlock-Studio uses **ONNX Runtime** with the **DirectML** provider (`DmlExecutionProvider`). This translates AI operations into **DirectX 12** calls, ensuring broad compatibility with NVIDIA, AMD, and Intel GPUs.

6.2. Tiling System and Memory Management

To handle high-resolution files, the application splits each frame into fragments (*tiles*). The size of these tiles is dynamically calculated using the **VRAM Limiter**. Additionally, Python's garbage collector (`gc.collect()`) is invoked to force memory release and ensure stability.

6.3. Resume and Checkpoint Functionality

If a video process is interrupted, the processed frames are saved. When restarting the task, the `check_video_upscaling_resume` function detects these files and resumes work from where it left off, saving time.

6.4. Asynchronous Frame Writing

During video upscaling, the frames processed by the GPU are sent to a separate writer thread. This allows the GPU to immediately start processing the next batch without waiting for the (slower) disk writing operation to finish, thus maximizing performance.

6.5. Frame Interpolation Pipeline

The RIFE models use a specialized interpolation pipeline that analyzes motion between frames to generate smooth intermediate frames. This enables higher frame rates or slow-motion effects with minimal artifacts.

6.6. Logging System and Diagnostics

Warlock-Studio implements a comprehensive logging system that includes:

- **Process Logs:** Record every stage of AI processing
- **Error Logs:** Capture detailed errors with stack traces
- **Performance Logs:** Measure processing times and resource usage
- **Debug Logs:** Detailed information for troubleshooting

6.7. SuperResolution-10 Model Optimization

The SuperResolution-10 model requires specific configurations:

- **CHW Preprocessing:** Conversion from image format (height, width, channels) to (channels, height, width)

- **Float32 Normalization:** Conversion to 32-bit floating point for maximum precision
- **Memory Management:** Use of contiguous arrays for memory access optimization
- **GPU Memory Allocation:** Dynamic VRAM allocation based on image size and available resources

6.8. Performance Monitoring and Metrics

The application provides detailed performance metrics:

Processing Speed:
Frames per second (FPS) and images per minute metrics

Memory Usage:
Real-time VRAM and system RAM monitoring

GPU Utilization:
DirectML provider performance statistics

Disk I/O:
Read/write speeds for temporary frame storage

7. Advanced Error Resolution and System Diagnostics

7.1. GPU Memory Management and VRAM Optimization

Warlock-Studio implements sophisticated GPU memory management:

Dynamic Tile Sizing:
Automatically calculates optimal tile sizes based on available VRAM

Memory Pool Management:
Pre-allocates and reuses memory buffers to reduce allocation overhead

Garbage Collection Integration:
Forces Python garbage collection at strategic points to free unused memory

VRAM Monitoring:
Real-time monitoring of GPU memory usage with automatic fallback to smaller tiles

7.2. Model Loading and Initialization Troubleshooting

> **Warning**
>
> Model loading failures are often caused by corrupted ONNX files, insufficient system permissions, or DirectML provider initialization errors.

Model File Corruption:
Verify model file integrity by checking file sizes against expected values

Provider Initialization Failures:
Check DirectML compatibility and ensure DirectX 12 is properly installed

Permission Issues:
Run application as administrator if model loading fails consistently

7.3. Video Processing Pipeline Diagnostics

The video processing pipeline consists of several stages that can be individually diagnosed:

1. **Frame Extraction:** Verify FFmpeg can read the input video format
2. **AI Processing:** Monitor VRAM usage and processing times per frame
3. **Frame Assembly:** Check for missing or corrupted intermediate frames
4. **Video Encoding:** Validate codec compatibility and hardware encoder availability

7.4. Performance Optimization Guidelines

Scenario	Recommended Settings
Low VRAM (< 4GB)	Input Resolution: 50%, Multithreading: OFF, Use RIFE_Lite
Medium VRAM (4-8GB)	Input Resolution: 75%, Multithreading: 2-4 threads, Standard models
High VRAM (> 8GB)	Input Resolution: 100%, Multithreading: 6-8 threads, High-quality models
SSD Storage	Keep frames: ON for faster resume, Use higher multithreading
HDD Storage	Keep frames: OFF to save space, Lower multithreading to reduce I/O

Table 4: Optimization settings based on system configuration.

7.5. Log File Analysis and Debugging

Warlock-Studio generates comprehensive logs in the Documents folder:

warlock_studio.log:
General application events, model loading, and processing status

error_log.txt:
Detailed error messages with Python stack traces

performance_log.txt:
Processing times, memory usage, and performance metrics

7.6. Common Error Patterns and Solutions

Error Pattern	Typical Cause	Solution Strategy
"DirectML device not found"	GPU drivers outdated or DirectX 12 not supported	Update GPU drivers, verify DirectX 12 compatibility
"ONNX Runtime initialization failed"	Corrupted model files or insufficient permissions	Re-download models, run as administrator
"FFmpeg process terminated"	Unsupported codec or corrupted input file	Convert input to supported format (MP4, H.264)
"Tile processing timeout"	GPU overheating or driver instability	Reduce tile size, check GPU temperature, update drivers
"Memory allocation failed"	System RAM exhausted	Close other applications, reduce multithreading

Table 5: Common error patterns and resolution strategies.