bloc97 / Anime4K Public

A High-Quality Real Time Upscaler for Anime Video

bloc97.github.io/anime4k/

MIT License

14.2k stars 1.2k forks

Star Notifications

Code Issues 63 Pull requests 5 Discussions Actions Projects Wiki Security

master Go to file

bloc97 ... on Nov 26, 2021

View code

README.md

Anime4K

Anime4K is a set of open-source, high-quality real-time anime upscaling/denoising algorithms that can be implemented in any programming language.

The simplicity and speed of Anime4K allows the user to watch upscaled anime in real time, as we believe in preserving original content and promoting freedom of choice for all anime fans. Re-encoding anime into 4K should be avoided as it is non-reversible, potentially damages original content by introducing artifacts, takes up to $O(n^2)$ more disk space and more importantly, does so without any meaningful decrease in entropy (lost information is lost).

Disclaimer: All art assets used are for demonstration and educational purposes. All rights are reserved to their original owners. If you (as a person or a company) own the art and do not wish it to be associated with this project, please contact us at anime4k.upscale@gmail.com and we will gladly take it down.

Foreword

Anime4K is optimized for **native 1080p anime encoded with h.264, h.265 or VC-1**.

Even if it might work, it is **not** optimized for downscaled 720p, 480p or standard definition anime (eg. DVDs). Older anime (especially pre-digital era production) have artifacts that are very difficult to remove, such as bad deinterlacing, camera blur during production, severe ringing, film grain, older MPEG compression artifacts, etc.

This is also not replacement for SRGANs, as they perform much better on low-resolution images or images with lots of degradation (albeit not in real time).

What Anime4K does provide is a way to upscale, in real time, 1080p anime for 4K screens while providing a similar *effect* to SRGANs and being much better than waifu2x (See comparisons).

Currently, research is being done on better real-time upscaling for lower resolution or older content.

v4.1 Low resolution experiment

Results from the experimental SRGAN shaders for 360p -> 4K: (zoom in to view details)

The images are sorted by algorithm speed, bicubic being the fastest. FSRCNNX and Anime4K are real-time while waifu2x and Real-ESRGAN are not.

v4

Installation Instructions for GLSL/MPV

We introduce a line reconstruction algorithm that aims to tackle the distribution shift problem seen in 1080p anime. In the wild anime exhibit a surprising amount of variance caused by low quality compositing due to budget and time constraints that traditional super-resolution algorithms cannot handle. GANs can implicitly encode this distribution shift but are slow to use and hard to train. Our algorithm explicitly corrects this distribution shift and allows traditional "MSE" SR algorithms to work with a wide variety of anime.

Source: https://fancaps.net/anime/picture.php?/14728493 | Mode: B

Bicubic
(1080p -> 4K)

Waifu2x
>1000ms

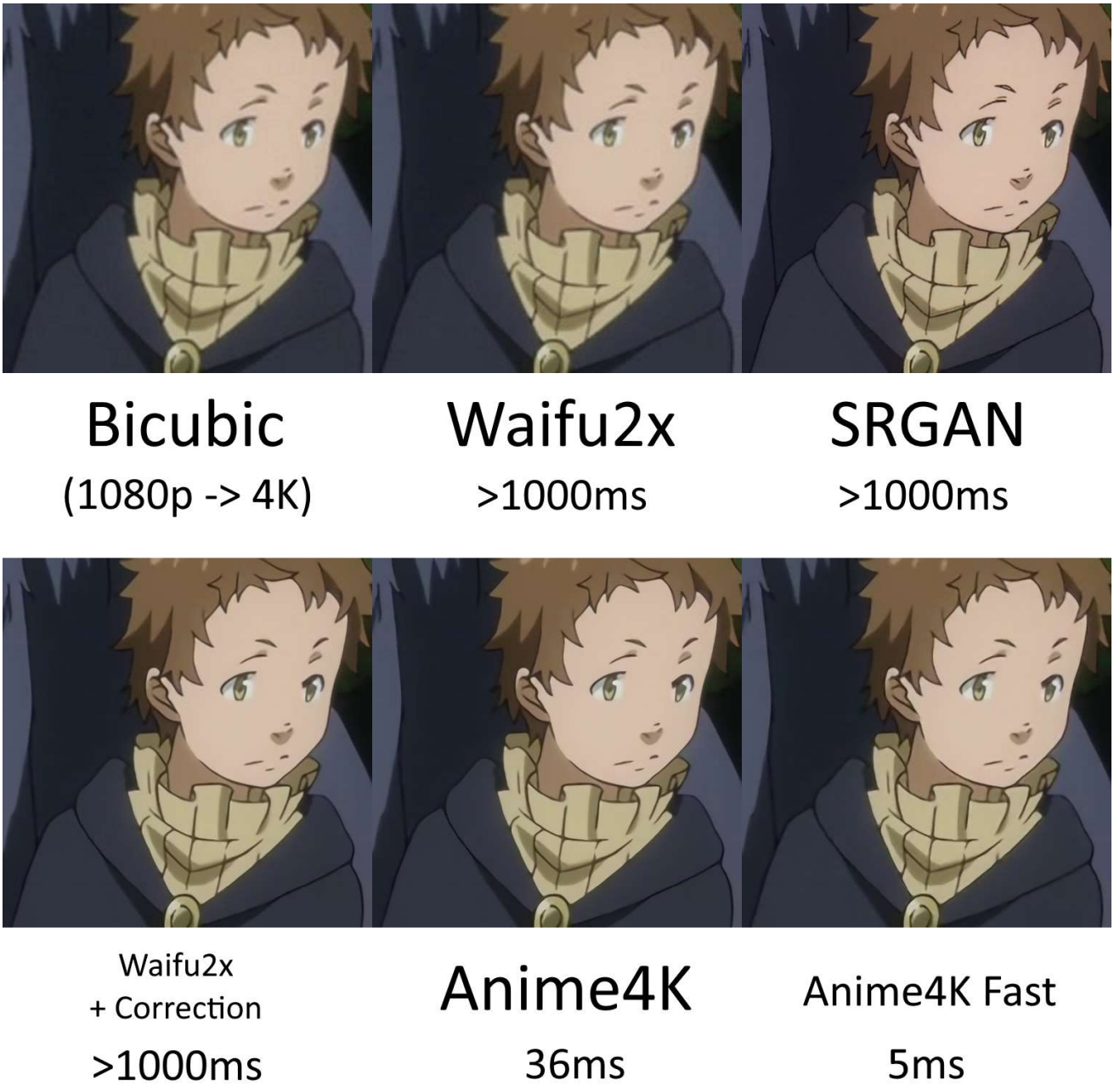
SRGAN
>1000ms

Waifu2x
+ Correction
>1000ms

Anime4K
36ms

Anime4K Fast
5ms

Source: https://fancaps.net/anime/picture.php?/13365760 | Mode: A



Performance numbers are obtained using a Vega64 GPU and are tested using `UL` shader variants. The fast version is for `M` variants.
Note that CUDA accelerated SRGANs/Waifu2x using tensor cores can be much faster and close to realtime (~80ms), but their large size severely hampers non-CUDA implementations.

v3

The monolithic Anime4K shader is broken into modular components, allowing customization for specific types of anime and/or personal taste. What's new:

- A complete overhaul of the algorithm(s) for speed, quality and efficiency.
- Real-time, high quality line art CNN upscalers. *(6 variants)*
- Line art deblurring shaders. *("blind deconvolution" and DTD shader)*
- Denoising algorithms. *(Bilateral Mode and CNN variants)*

- Blind resampling artifact reduction algorithms. *(For badly resampled anime.)*
- Experimental line darkening and line thinning algorithm. *(For perceptual quality. We perceive thinner/darker lines as perceptually higher quality, even if it might not be the case.)*

More information about each shader (OUTDATED).

Visits

Counting since `2021-09-19T16:02:06Z` (ISO 8601)

Projects that use Anime4K

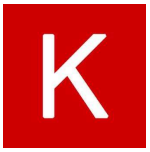
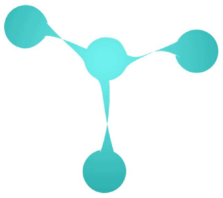
- https://github.com/Blinue/Magpie (General-purpose real-time upscaler for any program/game running on Windows 10)

Note that the following might be using an outdated version of Anime4K. There have been significant quality improvements since v3.

- https://github.com/yeataro/TD-Anime4K (Anime4K for TouchDesigner)
- https://github.com/keijiro/UnityAnime4K (Anime4K for Unity)
- https://github.com/net2cn/Anime4KSharp (Anime4K Re-Implemented in C#)
- https://github.com/andraantariksa/Anime4K-rs (Anime4K Re-Implemented in Rust)
- https://github.com/TianZerL/Anime4KCPP (Anime4K & more algorithms implemented in C++)
- https://github.com/k4yt3x/video2x (Anime Video Upscaling Pipeline)

Acknowledgements

OpenCV	TensorFlow	Keras	Torch	mpv	MPC

OpenCV	TensorFlow	Keras	Torch	mpv	MPC
					

Many thanks to the OpenCV, TensorFlow, Keras and Torch groups and contributors. This project would not have been possible without the existence of high quality, open source machine learning libraries.

I would also want to specially thank the creators of VDSR and FSRCNN, in addition to the open source projects waifu2x and FSRCNNX for sparking my interest in creating this project. I am also extending my gratitude to the contributors of mpv and MPC-HC/BE for their efforts on creating excellent media players with endless customization options. Furthermore, I want to thank the people who contributed to this project in any form, be it by reporting bugs, submitting suggestions, helping others' issues or submitting code. I will forever hold you in high regard.

I also wish to express my sincere gratitude to the people of Université de Montréal, DIRO, LIGUM and MILA for providing so many opportunities to students (including me), providing the necessary infrastructure and fostering an excellent learning environment.

I would also like to thank the greater open source community, in which the assortment of concrete examples and code were of great help.

Finally, but not least, infinite thanks to my family, friends and professors for providing financial, technical, social support and expertise for my ongoing learning journey during these hard times. Your help has been beyond description, really.

This list is not final, as the project is far from done. Any future acknowledgements will be promptly added.

Releases 16

GLSL (v4.0.1 Stable) Latest
on Sep 17, 2021

+ 15 releases

Packages

No packages published

Contributors 19

+ 8 contributors

Languages

- Jupyter Notebook 58.4%
- GLSL 41.4%
- Other 0.2%