zoomasm-3.3

Claude Heiland-Allen

2023-07-13

Contents

zoomasm

Zoom videos are a genre of 2D fractal animation. The rendering of the final video can be accelerated by computing exponentially spaced rings around the zoom center, before reprojecting to a sequence of flat images.

Some fractal software supports rendering EXR keyframes in exponential map form, which *zoomasm* can assemble into a zoom video. *zoomasm* works from EXR, including raw iteration data, and colouring algorithms can be written in OpenGL shader source code fragments.

Home: https://mathr.co.uk/zoomasm

Code: https://code.mathr.co.uk/zoomasm

Support: mailto:zoomasm@mathr.co.uk?subject=zoomasm

Donate: https://ko-fi.com/claudeha

Download: `zoomasm-3.3`

Example Videos

Tutorials

- Wurgo the Making Of with kf-2.15.2 and zoomasm-3.0 : https://archive.org/details/wurgo-the-making-of / https://diode.zone/videos/watch/5dd54244-90b4-423f-aae9-ae6545277c28 / https://www.youtube.com/watch?v=aTFtglO90T0

- Making fractal zoom videos with kf-2.15.1 and zoomasm-1.0 : https://archive.org/details/making-fractal-zoom-videos-with-kf-2.15-and-zoomasm-1.0 / https://diode.zone/videos/watch/d0b5f2f2-2aef-4405-92e6-4459a96ae93b / https://www.youtube.com/watch?v=72IIn7C3UeI

Music Videos

- Wurgo : https://archive.org/details/wurgo / https://diode.zone/videos/watch/5dd54244-90b4-423f-aae9-ae6545277c28 / https://www.youtube.com/watch?v=RpSpQ8kjRf0

- Charred Bard : https://archive.org/details/charred-bard / https://diode.zone/videos/watch/a121d1fb-baeb-45c4-a867-afeadac4f7fd / https://www.youtube.com/watch?v=NMKBBk-yf_4

- Special Branch : https://archive.org/details/special-branch / https://diode.zone/videos/watch/786b8320-b5aa-466f-a953-f6f814b59222 / https://www.youtube.com/watch?v=uQDV87vVIxk

- kf-exp-map-flat-1 : https://archive.org/details/kf-exp-map-flat-1

Prerequisites

- 64bit CPU and OS recommended:

    - Linux works (Debian Buster, Debian Bullseye, Debian Bookworm).
    - Windows not tested (but Wine works on Linux).
    - MacOS not tested.

- 8GB RAM recommended (works with 4GB at low resolutions).

- GPU and drivers supporting OpenGL 3.3 or later (OpenGL 4 is used when available).

- 8GB GPU VRAM recommended (works with 512MB at low resolutions).

- FFmpeg program binary.

Prepare Input

Using *Kalle's Fraktaler 2 +*

- Get KF https://mathr.co.uk/kf/kf.html#kf-2.15.

    KF version 2.15.1+ is required for exponential map export.

    KF version 2.15.2+ is required for OpenGL colouring.

- Launch it and zoom to where you want. It is recommended to enable **Derivatives** computation for analytic distance and slope colouring in *zoomasm*. Enabling **Jitter** is also recommended to avoid Moiré artifacts. Disabling any distance and slope colouring in KF is recommended if you want to use the R G B channels with KF's colours.

- Set the **Window Size** in KF to 1152x128 (aspect ratio 9:1).

- Set **Image Size** in KF to correspond to desired final video output resolution for 2D projection in *zoomasm*:

    - 320x180 / 1152x128

- 640x360 / 2304x256
- 1280x720 / 4608x512
- 1920x1080 / 6912x768
- 3840x2160 / 12288x1360 (window size 1536x170)

These are minimum recommendations, going bigger will give a higher quality final output video (you can go smaller too, for lower quality to save time).

Recommended image sizes for 360 projection in *zoomasm*:

- 1024x512 / 1024x112
- 2048x1024 / 2048x224 (window size 1024x112)
- 4096x2048 / 4096x448 (window size 1024x112)
- 8192x4096 / 8192x912 (window size 1024x114)

The maximum keyframe size in *zoomasm* is limited by GPU capabilities, it is recommended to do a small test first to check that *zoomasm* will be able to handle your keyframes on your hardware. Check that you can fit the desired number of layers into video memory, for 2D projection 16 is ok, for 360 projection as many as 25 or more may be needed to avoid artifacts at the poles.

- Activate **Exponential Map** in the **View** menu. You may use `PageUp` and `PageDown` to fine tune final zoom level. You want the bottom edge of the image to be a uniform colour (most often the interior colour).

- Choose which **EXR Channels** you want to save.

- **Store the zoom out sequence** (be sure to save EXR!).

- Wait while it renders (this is often the slow part of the process). You may use the about dialog (**?** menu) to block accidental input. Alternatively you can render from the command line without any window at all.

- Quit KF.

Soundtrack

- Prepare your soundtrack and save as WAV or FLAC. Alternatively you can work with silence (specify duration).

User Guide

Quick Start

- Select **Audio**/**Silence** duration or **Audio**/**Soundtrack** audio file (top left drop-down combo box widget).
- Select **Input**/**Input** keyframe directory.
- Select **Colour**/**Colour** and choose one of the `presets/*.glsl`. If a blank image results, pick a different one or check which image channels they need. You may also load a `*.kfp` or `*.kfr` palette with OpenGL GLSL colouring, as saved by Kalle's Fraktaler version 2.15.2 or later.
- If **Output**/**FFmpeg** version is not detected, choose your `ffmpeg` (`ffmpeg.exe` on Windows).
- Select **Output**/**Output** video file (`output.mp4`). Select **Output**/**Overwrite** if the file already exists and you want to overwrite it.
- Select **Output**/**Record** to render the zoom video.
- Wait until it finishes rendering.
- Play `output.mp4` in VLC or similar.

Session

- **Save** saves session to TOML text file (`*.toml`).

Figure 1: Screenshot of zoomasm example session.

- **Load** loads session from TOML text file (`*.toml`). **WARNING** do not load session files from untrusted sources, because the path to the FFmpeg binary is stored in the file, which is invoked on load to get FFmpeg version information.
- **Version** displays program version information.
- **Source** exports this *zoomasm*'s source code archive (`*.7z`).

Audio

- **Silence** combo box option to work without audio. Specify duration in seconds. No audio track will be present in the output video file.
- **Soundtrack** combo box option to choose audio file (`*.wav`, `*.flac`). Detected sample rate, channels and duration are displayed below.
- **Time** slider bar displays current time and allows seeking in the input.
- **Rewind** button resets time to `0.000s` (start of file).
- **Stop** button turns off automatic advance.
- **Play** button turns on automatic advance.
- **Loop** checkbox activates loop play mode.
- **1x** button resets playback speed to `1.000x`.
- **Speed** slider sets playback speed (reverse and freeze are possible, *zoomasm* uses a phase vocoder for time warping).
- **Mute** checkbox silences output.
- **Volume** slider sets playback volume.
- Counts of any realtime errors (ring buffer overruns, silence due to sample data not being loaded in time) are displayed below, with a **Reset** button to clear.

Input

- **Input** button to choose directory containing exponential map strip keyframes (`*.exr`).
- Detected count and resolution are displayed.

- **Channels** checkboxes choose which channels from the files to upload to the GPU. Defaults to all channels if there is enough VRAM, no channels otherwise.
- An estimate of GPU VRAM (video memory) usage is displayed.
- **Override** checkbox forces all selected channels to be uploaded, even if there is not enough video memory detected.
- **Reverse** checkbox should be activated if keyframes are stored in zoom-out order (like KF).
- **Flip** checkbox flips each keyframe vertically (for compatibility with other software).
- **Invert** checkbox inverts zoom projection (for special effects).
- **Layers** number box to control how many keyframes are loaded into video memory. 16 is good for 2D projection, 25 for 360 projection. Increase it if you see small circles near the zoom focus.

Colour

- **Colour** button to choose colouring OpenGL GLSL shader fragment (`*.glsl`) or Kalle's Fraktaler palette (`*.kfp`, `*.kfr`) (kf-2.15.2 or later). Compilation status is displayed below including compilation error log. Examples in the `presets/` folder include:

  - `binary-decomposition.glsl`: works best with linear smoothing and a custom bailout radius of 25. Uses `N` and `T` image channels. Needs a high sample count not to look awful.

  - `cycling-waves.glsl`: colour cycling demo. Uses `N` and `NF` image channels for the base colouring, as well as `DEX` and `DEY` for slope lighting effect. Uses `getTime()` to animate colour cycling.

  - `multiwave.glsl`: randomized fractal colour palettes based on an idea by Pauldelbrot. Use `N`, `NF`, `DEX`, `DEY`. Choose the seed value for the pseudo-random number generator at the top of the file.

  - `rainbow-fringe.glsl`: simple black on white distance estimate with surrounding rainbows based on the DE direction. Uses the `DEX` and `DEY` image channels.

  - `rgb-passthrough.glsl`: just passes through the `R`, `G`, `B` image channels. Tends to look bad if the source keyframes used distance estimates or slopes in their colouring algorithm as the details near the center will be smaller than the edges (*zoomasm* has special compensation for the `DEX` and `DEY` channels).

  - `text-overlay.glsl`: same as the previous, but with text overlay using `getZoomLog2()` for zoom depth in 2x zoom levels, displayed in the top-right corner (assumes 16:9 aspect ratio).

  WARNING do not load GLSL or KFP/KFR from untrusted sources, there is a possibility to crash graphics drivers if the shader has an infinite loop or similar.

- **Watch** checkbox to reload shader automatically when it is changed (hit save in your favourite text editor, see live colouring changes). The last successfully compiled shader is kept instead of displaying a blank image on errors (this is better for live coding).

- **Samples per pixel** can be increased for better quality (smoother appearance in the center, less "sparkling" during animation) but at higher computational cost (on the GPU).

- **Shutter speed** can be adjusted for motion blur, between 0% and 100% of frame duration (see **Output**/**FPS**).

- **Projection** offers a choice of video remapping options:

  - `2D` is the most commonly used projection for zoom videos, it has a central focus and a rotation angle control.

  - `360` is an equirectangular projection, for surround viewing with 3D rotation control (axis and angle). You can inject 360 metadata into encoded MP4 video files with the spatial media tool: https://github.com/google/spatial-media

Timeline

- **Import** button to import timeline from CSV.
- **Export** button to export timeline to CSV.
- **Headerless** checkbox to import/export CSV without a header line.
- **Interpolated** checkbox to export interpolated timeline (cannot be imported back into *zoomasm*). Set **Output**/**FPS** for resolution control.

Waypoints in the zoom animation are shown below.

The first (at the start of the soundtrack) and last (at the end of the soundtrack) cannot be deleted, the rest have a **-** button to remove them (permanently, no undo). If the **Audio**/**Time** slider is between two waypoints, a cursor appears in red, pressing its **+** button adds a new waypoint at that time.

Each waypoint has a **Time** button at the left, clicking this button jumps to that time in the audio playback. Each waypoint has a **Z** slider to set zoom depth (in terms of input EXR keyframe index) at the corresponding time (displayed on the left). The interpolation between waypoints is set by the **Interpolation** dropdown combo box, defaulting to (monotone) **Cubic** for smoother speed changes (other options are **Step** and **Linear**). Instantaneous zoom speed in 2x and 10x zooms per second is displayed below the cursor.

Output

- **Record** button launches the video encoder, with a modal dialog blocking input with a progress bar and **Stop** button. The Record button is only displayed when recording is possible (hint: the widgets in red need to be completed). The recording speed is not limited to display vsync (display updates target 20fps). Time elapsed and ETA to completion is displayed.
- **Status** displays current application update frame rate.
- **FFmpeg** button to choose `ffmpeg` (or `ffmpeg.exe` on Windows) program binary. FFmpeg is needed for all formats apart from PPM image sequence. Detected `ffmpeg` version is displayed below.
- **Output** button to choose output video file (`*.mp4`, `*.mkv`, `*.mov`, `*.ppm`). Saving to `foo-.ppm` will write numbered frames starting `foo-00000000.ppm`, `foo-00000001.ppm`, etc. PPM is a simple lossless uncompressed binary bitmap image format with an ASCII metadata header at the start. It will take up a *lot* of space (21GB/min at 1920x1080p60) so it is recommended to encode directly with FFmpeg if possible, using any of the other formats.
- **Overwrite** checkbox to allow overwriting an existing file.
- **Resolution** dropdown (with custom size option) to set video frame size. Information on anti-aliasing level is displayed below (higher texels per pixel is better).
- **FPS** dropdown (with custom framerate option) to set video frames per second.
- **Video CRF** number box to set video encoder Constant Rate Factor quality (0 is lossless/incompatible, 1 is best quality/largest size, 51 is lowest quality/smallest size). Changing the value by 6 will change the file size by a factor of 2, approximately.
- **Audio Bitrate** dropdown (with custom bitrate option) to set audio bitrate in kbps. The special option **Copy** attempts to put the soundtrack into output video file without transcoding (be sure to check encoder messages to see if it worked).
- **Encoder log** contains video encoder messags.
- **Advanced** checkbox allows full control over FFmpeg command line options, including two-pass encoding.

Keyboard Controls

When no widget is active:

- Press `Space` to toggle pause/play (when not recording).
- Press `Backspace` to reverse playback speed.
- Press `Enter` or `Return` to add a new waypoint at the current time.
- Press `F9` to toggle the user interface transparency.
- Press `F10` to toggle the user interface visibility.
- Press `F11` to toggle fullscreen (only on primary monitor so far).

You can press `Tab` to navigate between widgets, and `Ctrl`-clicking a slider allows to enter numbers more precisely.

Command Line

- `--help` (`-h`, `-H`, `-?`) displays *zoomasm* usage information.
- `--version` (`-v`, `-V`) displays version information of *zoomasm* and its supporting libraries.
- `--source` (`-S`) exports *zoomasm*'s source code archive to a file in the current working directory.
- `--fullscreen` (`-f`) starts in fullscreen mode (on primary monitor).
- `--no-fullscreen` (`-F`) starts in windowed mode (default).
- `--gui` (`-g`) show GUI on startup (default).
- `--no-gui` (`-G`) hide GUI on startup.
- `--record` (`-r`) activates recording on startup and quits when it has finished. Requires a session filename to do anything useful.
- `--no-record` (`-R`) normal interactive mode (default).
- One saved session filename (`*.toml`) can be added to the command line, which will be loaded before anything else happens.

GLSL API

For full details see `src/main_frag.glsl` in the source distribution.

The GLSL shader (whether standalone or embedded in KFP/KFR) needs to define a function `vec3 colour(void)` that returns linear-light RGB. For best results set **Use sRGB** on in *KF* to automatically convert the palette from sRGB to linear in *zoomasm*

getInterior()

Check if the pixel is unescaped interior.

getGlitch()

Check if the pixel is unevaluated or glitched or otherwise bad.

getN()

Gets the smooth iteration count. Only valid for non-glitch non-interior pixels. Returns a `float49` type with higher precision than the 24 bits available in single precision `float`. Due to lack of operator overloading in GLSL, you need to use functions like `add()`, `sub()` etc instead of usual maths. At the end you can use `to_float()` when higher precision is no longer needed.

The raw channels can be retrieved with `getN0()`, `getN1()`, `getNF()`. They are biased with an offset to avoid negative values. The offset is in the uniform variable `IterationsBias`.

getT()

Get the phase channel.

getDE()

Get the distance estimate (`DEX` and `DEY`). *zoomasm* scales this so that it is relative to screenspace no matter the projection.

getRGB()

Get the colour stored in the input keyframes.

`getCoord()`

Get output coordinates, relative to the uniform variable `ImageSize`. Origin is bottom left per OpenGL conventions.

`KF_Colour()`

Colour the pixel emulating KF's KFP/KFR colouring. Parts of the algorithm are modularized, check KF's user manual for details.

Build Guide

From Linux

Prerequisites

Whether you are building for Windows or for Linux, you need:

```
sudo apt install git wget xz-utils build-essential p7zip pandoc texlive-latex-recommended
```

For Linux

Prerequisites

- Debian Bullseye (current oldstable) works, but newer is recommended.
- Depending on compiler version you might need to add `EXTRA_LIBS="-lstdc++fs"` to the `make` build command.
- You need an OpenEXR version compatible with C++17; Debian Buster's version is too old.

Build

See first the general prerequisites.

```
sudo apt install git libglew-dev libglfw3-dev libz-dev libopenexr-dev libfftw3-dev ffmpeg pkg-config
mkdir z
cd z
git clone https://code.mathr.co.uk/zoomasm.git
git clone https://github.com/ocornut/imgui.git
git clone https://github.com/AirGuanZ/imgui-filebrowser.git
git clone https://github.com/mackron/miniaudio.git
git clone https://github.com/marzer/tomlplusplus.git
cd zoomasm
make zoomasm
```

Tested with versions as of 2023-07-13:

- imgui `v1.89.7-18-g77eba4d0d`
- imgui-filebrowser `fbafb08`
- miniaudio `0.11.17`
- tomlplusplus `v3.3.0-8-g4a28c36`

Run

```
./zoomasm
```

For Windows From Linux

Prerequisites

See first the general prerequisites.

- Debian Bullseye (current oldstable) or newer is recommended.
- Install the compiler and runtime:

```
sudo apt install mingw-w64 wine64 cmake unzip
```

- Depending on compiler version you might need to add `-lstdc++fs` to the build commands in the `Makefile`.

- MINGW64 POSIX threading model is needed for C++ `std::thread` etc, so configure that globally (you can ignore failures for gfortran and gnat if they are not installed):

```
update-alternatives --set x86_64-w64-mingw32-g++ /usr/bin/x86_64-w64-mingw32-g++-posix
update-alternatives --set x86_64-w64-mingw32-gcc /usr/bin/x86_64-w64-mingw32-gcc-posix
update-alternatives --set x86_64-w64-mingw32-gfortran /usr/bin/x86_64-w64-mingw32-gfortran-posix
update-alternatives --set x86_64-w64-mingw32-gnat /usr/bin/x86_64-w64-mingw32-gnat-posix
update-alternatives --set i686-w64-mingw32-g++ /usr/bin/i686-w64-mingw32-g++-posix
update-alternatives --set i686-w64-mingw32-gcc /usr/bin/i686-w64-mingw32-gcc-posix
update-alternatives --set i686-w64-mingw32-gfortran /usr/bin/i686-w64-mingw32-gfortran-posix
update-alternatives --set i686-w64-mingw32-gnat /usr/bin/i686-w64-mingw32-gnat-posix
```

- *zoomasm* expects dependencies to be installed into `~/opt/windows/posix/x86_64/` (or `~/opt/windows/posix/i686/` for 32bit).

- You can install the dependencies using https://mathr.co.uk/web/build-scripts.html:

```
git clone https://code.mathr.co.uk/build-scripts.git
cd build-scripts
./BUILD.sh download "zlib imath openexr3 fftw3 glfw glew"
./BUILD.sh x86_64-w64-mingw32 "zlib imath openexr3 fftw3 glfw glew"
./BUILD.sh i686-w64-mingw32 "zlib imath openexr3 fftw3 glfw glew"
```

Build

```
mkdir -p ~/opt/src
cd ~/opt/src
git clone https://code.mathr.co.uk/zoomasm.git
git clone https://github.com/ocornut/imgui.git
git clone https://github.com/AirGuanZ/imgui-filebrowser.git
git clone https://github.com/mackron/miniaudio.git
git clone https://github.com/marzer/tomlplusplus.git
cd zoomasm
make zoomasm.i686.exe
make zoomasm.x86_64.exe
```

Tested with versions as of 2023-07-13:

- imgui `v1.89.7-18-g77eba4d0d`
- imgui-filebrowser `fbafb08`
- miniaudio `0.11.17`
- tomlplusplus `v3.3.0-8-g4a28c36`

Run

```
wine zoomasm.i686.exe
wine zoomasm.x86_64.exe
```

To encode videos, `zoomasm.exe` needs `ffmpeg.exe`. Builds are available via: https://ffmpeg.org/download.html#build-windows.

To Do

Audio

- Sample-accurate looping with read-ahead for the other side of the loop.
- Investigate zero-padding for FFT (might improve timestretch quality?).
- Don't close and reopen device if channels/samplerate didn't change.
- Soundtrack generation from input EXR data.
- Soundtrack generation from output RGB data.
- Switch from GLFW to SDL2 and drop miniaudio dependency.

Input

- Support `N0 + N1` channels as well as just `N`; for very high iteration counts.
- Handle errors more gracefully (eg currently input/output error will exit *zoomasm* abruptly).

Colour

- Built-in text-editor for colouring shader fragment, with save/load to global session settings and export/import to separate `.glsl` file.
- Detect active non-internal uniforms in the colouring shader fragment and expose controls in the GUI.
- Image (and video) texture input via FFmpeg.

Timeline

- Sequence custom uniform values.
- Lua (or other embedded language) scripting for controlling uniforms.
- Make timestamps into buttons to jump to time.
- Set instantaneous speed at waypoints.
- Add exponential (in terms of speed change) interpolation mode.

Miscellaneous

- Display OpenGL debug log in program.
- Improve `Makefile` to support parallel builds and object reuse.
- Make `Enter` key work when display is hidden.

Hacking

zoomasm/

- `COPYING.md`: GNU Affero General Public License.
- `README.md`: documentation.
- `INDEX.txt`: list of files to include in source code release bundles. Remember to update when adding/removing files, and check that the source code release bundles build correctly.
- `VERSION.txt`: current *zoomasm* version. Should not be checked into version control (it is regenerated from git version information).
- `zoomasm.css`: style sheet for HTML website.
- `zoomasm.png`: screenshot for documentation.
- `Makefile`: build system.

`zoomasm/src/`

- `s2c.sh`: build system helper (text file contents to C source code).
- `audio.cc`, `audio.h`: audio playback and realtime scheduling.
- `colour.cc`, `colour.h`, `main_frag.glsl`, `main_vert.glsl`: colouring shader management.
- `ffmpeg.cc`, `ffmpeg.h`: interface to FFmpeg process.
- `input.cc`, `input.h`: input EXR keyframe loading.
- `main.cc`: main program
- `output.cc`, `output.h`: recording from OpenGL, and encoder setttings.
- `process.h`, `process_posix.cc`, `process_win32.cc`: abstraction for launching subprocesses with bidirectional communication on different operating systems.
- `resource.h`: common constants.
- `session.cc`, `session.h`: session management, version information and source code archive export.
- `timeline.cc`, `timeline.h`: waypoint sequencing.
- `utility.cc`, `utility.h`: miscellaneous utilities.

Tips

- Prevent breakage when writing to FFmpeg pipe fails in `gdb`:

```
handle SIGPIPE nostop noprint pass
```

Releases

Releases are signed with GPG key id `EC470ECD90DDE39B6ED67CA6EBC1FED7E3FA39B0`.

3.3

- Codename: "igualment"
- Released: 2023-07-13
- Source: `zoomasm-3.3.7z` (sig)
- Windows: `zoomasm-3.3-win.7z` (sig)
- Manual: HTML PDF
- Changes:
    - Fix: correct shader version syntax (thanks Chris Hubble).
    - Fix: clarify prerequisites documentation.
    - Win: build dependencies using build-scripts
    - Win: change library prefix to `~/opt/windows/posix/$arch`.
    - Win: update `openexr` to 3.1.9.
    - Win: update `Imath` to 3.1.9.
    - Win: update `imgui`, `imgui-filebrowser`, `miniaudio`, and `tomlplusplus` to latest versions.

3.2

- Codename: "molt de gust"
- Released: 2022-12-19
- Source: `zoomasm-3.2.7z` (sig)
- Windows: `zoomasm-3.2-win.7z` (sig)
- Manual: HTML PDF
- Changes:
    - Fix: updated to current `imgui` API (v1.89).
    - Fix: support `openexr` 3.1.5.
    - New: can use `make EXTRA_LIBS="-lstdc++fs" zoomasm`, no longer need to edit `Makefile`.
    - Win: update `zlib` to 1.2.13.
    - Win: update `openexr` to 3.1.5.

- – Win: update `Imath` to 3.1.6.
- – Win: update `imgui`, `imgui-filebrowser`, `miniaudio`, and `tomlplusplus` to latest versions.
- – Win: patch `tomlplusplus` to revert `cc741c9` due to build failure.

3.1

- Codename: "d'acord"
- Released: 2022-08-03
- Source: `zoomasm-3.1.7z` (sig)
- Windows: `zoomasm-3.1-win.7z` (sig)
- Manual: HTML PDF
- Changes:
  - – API: shaders can use `#if ZOOMASM_VERSION_MIN(3,1)` to adapt to actual *zoomasm* version.
  - – Fix: no longer crashes when loading session with directory that does not exist.
  - – Fix: updated to current `miniaudio` API (0.11.9).
  - – Fix: prefer PulseAudio to JACK (workaround issue with PipeWire).
  - – Win: change library prefix to `~/win/posix/$arch`.
  - – Win: update `zlib` to 1.2.12.
  - – Win: update `openexr` to 2.5.8.
  - – Win: update `fftw` to 3.3.10.
  - – Win: update `glfw` to 3.3.8.
  - – Win: update `imgui`, `imgui-filebrowser`, `miniaudio`, and `tomlplusplus` to latest versions.

3.0

- Codename: "som-hi tots"
- Released: 2021-04-02
- Source: `zoomasm-3.0.7z` (sig)
- Windows: `zoomasm-3.0-win.7z` (sig)
- Manual: HTML PDF
- Changes:
  - – API: backwards incompatible change: GLSL shaders now need to define `vec3 colour(void)` and call `getX()` functions instead of having everything as arguments. See the manual for API documentation.
  - – New: support for KFR/KFP palettes with OpenGL GLSL shaders from *Kalle's Frakaler 2 +* version 2.15.2 or later. Some features are missing:
    - * Iterations, IterationsMin, IterationsMax;
    - * Jitter compensation for numerical differences-based DE;
    - * Texture images.
  - – Win: change binary names to x86_64 and i686: maybe aarch64 or armv7 will be possible later using llvm-mingw.
  - – Win: put `presets/` in a subfolder of the distribution
  - – Win: update `openexr` to 2.5.5
  - – Win: update `glew` to 2.2.0
  - – Win: update `glfw` to 3.3.3
  - – Win: update `imgui`, `imgui-filebrowser`, `miniaudio`, and `tomlplusplus` to latest versions.

2.1

- Codename: "tot bé"
- Released: 2021-02-02
- Source: `zoomasm-2.1.7z` (sig)
- Windows: `zoomasm-2.1-win.7z` (sig)
- Manual: HTML PDF

- Changes:
  - New: timeline waypoint timestamps are now buttons that jump to time.
  - Fix: the `Z` of the `XYZT` colouring parameter is now per-pixel.
  - Fix: correct scaling of `DE` in 360 projection.
  - Fix: vanishing custom size widgets.
  - Fix: compiles with recent `imgui`.
  - Fix: typo in make clean target.
  - Win: change dependency build directory to `~/win/64/posix`.
  - Win: update `openexr` to 2.5.4
  - Win: update `fftw` to 3.3.9
  - Win: update `imgui`, `imgui-filebrowser`, `miniaudio`, and `tomlplusplus` to latest versions.

2.0

- Codename: "bon profit"
- Released: 2020-12-08
- Source: `zoomasm-2.0.7z` (sig)
- Windows: `zoomasm-2.0-win.7z` (sig)
- Manual: HTML PDF
- Changes:
  - New: `multiwave.glsl` example colouring preset.
  - New: 360 equirectangular projection. Regular 2D projection is still available. Both have rotation controls (in 3D for 360 projection).
  - New: speculatively load next keyframe in async thread for smoother interactive playback.
  - New: advanced mode allows setting arbitrary FFmpeg encoder options.
  - New: two pass encoding (in advanced mode only).
  - Fix: use conservative latency profile for audio.
  - Fix: prefer JACK (if available) to PulseAudio.
  - Fix: be more robust when changing soundtracks.
  - Fix: try to detect death spiral and stop playback if keyframe loading can't keep up with real time.
  - Fix: recycle audio buffer memory instead of freeing and allocating so often.
  - Fix: disable broken MP3 support (no seeking with reverse play).
  - Fix: disable broken Ogg support (no duration detection).
  - Fix: only rerender frame if something changed (reduces GPU load when stopped).
  - Fix: use `texelFetch()` instead of `texture()` (prevents flickering boxes which might have been an amdgpu driver bug).
  - Fix: loading session from command line does not reset last waypoint.
  - Fix: allow manual control over number of keyframe layers at runtime (16 is ok for 2D, but 360 projection can need more, try 25 or so).
  - Fix: avoid OpenGL error clearing an incomplete framebuffer object.
  - Fix: remove unnecessary OpenGL sync objects.
  - Fix: don't use features newer than OpenGL 3.0 (but shaders still need OpenGL 3.3 with GLSL 330 core profile, or OpenGL 4.0 with GLSL 400 core profile).
  - Fix: enable OpenEXR multithreading.
  - Fix: combined single-file build system.
  - Win: rewrite subprocess module to use `_spawnv()`.
  - Win: build both 64bit and 32bit Windows EXEs.

1.1

- Codename: "salut"
- Released: 2020-11-08
- Source: `zoomasm-1.1.7z` (sig)
- Windows: `zoomasm-1.1-win.7z` (sig)

- Manual: HTML PDF
- Changes:
    - New: `F9` key toggles user interface transparency.
    - New: `.ppm` output file saves an image sequence without FFmpeg.
    - Fix: no more garbage audio output for an instant on device open.
    - Fix: fix build with recent upstream `imgui` changes.
    - Fix: when loading a session, don't reset the last timeline waypoint zoom depth to the default for the input keyframe count.
    - Fix: reduce timeline Z slider range to avoid glitches at extremes.
    - Fix: add an example preset that was missing from the distribution.
    - Doc: add demo/tutorial videos to documentation.
    - Doc: round recommended keyframe image sizes to multiples of 16 so that `exrsubsample` can be used without worrying about edge effects.
    - Win: include `presets/` folder in Windows distribution.
    - Win: include PDF manual in Windows distribution.
    - Win: update `imgui`, `miniaudio`, `tomlplusplus` to latest versions. `miniaudio` needed a small patch to build as C++ `void*` casts must be explicit (unlike as in C); this will hopefully be fixed upstream soon.

1.0

- Codename: "felicitats"
- Released: 2020-10-07
- Source: `zoomasm-1.0.7z` (sig)
- Windows: `zoomasm-1.0.exe.7z` (sig)
- Manual: HTML PDF
- Changes:
    - First released version.

Legal

zoomasm

zoomasm – zoom video assembler

Copyright (C) 2019,2020,2021,2022,2023 Claude Heiland-Allen

This program is free software: you can redistribute it and/or modify it under the terms of the GNU Affero General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Affero General Public License for more details.

You should have received a copy of the GNU Affero General Public License along with this program. If not, see https://www.gnu.org/licenses/.

FFTW

FFTW is used under GPL license (version 2 or later) http://fftw.org/

FFTW is Copyright © 2003, 2007-11 Matteo Frigo, Copyright © 2003, 2007-11 Massachusetts Institute of Technology.

FFTW is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA. You can also find the GPL on the GNU web site: http://www.gnu.org/licenses/gpl-2.0.html

GLEW

GLEW is used under the Modified BSD License, the Mesa 3-D License (MIT) and the Khronos License (MIT). http://glew.sourceforge.net/

The OpenGL Extension Wrangler Library

Copyright (C) 2008-2016, Nigel Stewart <nigels[]users sourceforge net>

Copyright (C) 2002-2008, Milan Ikits <milan ikits[]ieee org>

Copyright (C) 2002-2008, Marcelo E. Magallon <mmagallo[]debian org>

Copyright (C) 2002, Lev Povalahev

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- The name of the author may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Mesa 3-D graphics library

Version: 7.0

Copyright (C) 1999-2007 Brian Paul All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense,

and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL BRIAN PAUL BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright (c) 2007 The Khronos Group Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and/or associated documentation files (the "Materials"), to deal in the Materials without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Materials, and to permit persons to whom the Materials are furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Materials.

THE MATERIALS ARE PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE MATERIALS OR THE USE OR OTHER DEALINGS IN THE MATERIALS.

GLFW

GLFW is used under zlib/libpng license (BSD-style) https://www.glfw.org/

Copyright © 2002-2006 Marcus Geelnard

Copyright © 2006-2019 Camilla Löwy

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

- The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

- Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

- This notice may not be removed or altered from any source distribution.

imgui

imgui is used under MIT license (with public domain portions) https://www.dearimgui.org/

The MIT License (MIT)

imgui-filebrowser

imgui-filebrowser is used under MIT license https://github.com/AirGuanZ/imgui-filebrowser

miniaudio

miniaudio is used under MIT-0 license (or public domain) https://miniaud.io/

dedication to be an overt act of relinquishment in perpetuity of all present and future rights to this software under copyright law.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

For more information, please refer to http://unlicense.org/

ALTERNATIVE 2 - MIT No Attribution

Copyright 2020 David Reid

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

OpenEXR

OpenEXR is used under OpenEXR license (BSD-style) https://www.openexr.com/

Copyright Contributors to the OpenEXR Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT

(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

QD

QD (portions ported from C++ to GLSL) is used under the QD license:

1. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

   (1) Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

   (2) Redistributions in binary form must reproduce the copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

   (3) Neither the name of the University of California, Lawrence Berkeley National Laboratory, U.S. Dept. of Energy nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

2. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

3. You are under no obligation whatsoever to provide any bug fixes, patches, or upgrades to the features, functionality or performance of the source code ("Enhancements") to anyone; however, if you choose to make your Enhancements available either publicly, or directly to Lawrence Berkeley National Laboratory, without imposing a separate written license agreement for such Enhancements, then you hereby grant the following license: a non-exclusive, royalty-free perpetual license to install, use, modify, prepare derivative works, incorporate into other computer software, distribute, and sublicense such enhancements or derivative works thereof, in binary and source code form.

toml++

toml++ is used under MIT license https://github.com/marzer/tomlplusplus

MIT License

Copyright (c) Mark Gillard <mark.gillard[]outlook.com.au>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

zlib

zlib is used under zlib license (BSD-style) https://zlib.net/

Copyright (C) 1995-2017 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly Mark Adler

jloup[]gzip.org madler[]alumni.caltech.edu

https://mathr.co.uk