

5DtoRGB 1.5.4 Preview (Windows)

August 4, 2011

Copyright © 2010-2011 Rarevision LLC

<http://rarevision.com/5dtorgb/>

support@rarevision.com

New Features

As of version 1.5, 5DtoRGB uses a new, OpenGL-based transcoding engine which offloads the transcoding task to your video card (GPU). It is both faster and higher-quality than the previous engine. Please note that the new engine requires OpenGL support from the operating system. Most modern video cards support hardware-accelerated OpenGL, so this shouldn't be a problem for most users. However, out-of-date video card drivers or older video cards may produce color distortions in 5DtoRGB's output and/or cause the program to crash. If you encounter these problems, try updating your video card's driver.

Basic Usage

5DtoRGB transcodes files in the AVC/H.264 video format to either QuickTime or AVI files in a variety of codec / picture formats.

To convert a file, do this:

1. Click the "Convert..." button in the main window or select "Open..." from the "File" menu.
2. In the file browser window, select the file you would like to convert and click "OK."
3. Specify an output format, decoding matrix and program options, then click "Convert..."
4. Use the file browser to select a save location for the output file.
5. Once the progress bar reaches 100%, the conversion is complete!

NOTE: Clicking "Cancel" will terminate the conversion process. If a QuickTime or AVI conversion is in progress, 5DtoRGB will remove the partially converted file.

Program Options

Output Format - Choose the output format to which you want to transcode. Selecting a QuickTime (MOV) option will output a QuickTime file with embedded timecode. Selecting an AVI option will output an AVI file without embedded timecode. The following codecs / picture formats are supported:

Uncompressed 8 bit 4:2:2 (UYVY)

Uncompressed 10 bit 4:2:2 (v210)

Avid DNxHD (common bit rates supported)

The following are lossless RGB formats, suitable for visual effects work:

QuickTime Animation (RLE)

HuffYUV (typically used with AVI format)

Frame Rate Selection - Use this to set your footage play at an arbitrary frame rate. This is useful for "overcranking" effects, like when shooting at 60 fps for 24 fps output. You can also use this for "undercranking" effects or any other special purpose. Keep in mind that this does not "convert"

footage from one frame rate to another. It wouldn't be useful for converting 29.97 fps material to 23.976, for example. It simply sets the playback rate of the video without changing the frame count.

Decoding Matrix - Choose the decoding matrix you want to use here. Normally, you would match the matrix that was used to encode the video to YUV (technically YCbCr). Generally speaking, BT.709 is for HD material and BT.601 is for SD. Canon HDSLRs use the BT.601 matrix, which is selected by default.

Gamma Flagging - This setting will add a gamma flag to QuickTime files without actually affecting the picture data (pixel values) in the footage. This flag can be read by QuickTime-enabled applications, which may or may not use this setting to affect the way your footage is displayed. If you notice that your footage looks either too light or too dark when importing it into a QuickTime-enabled app, try changing this setting.

Timecode Support

5DtoRGB will derive start timecode values from the timecode value entered into the "Start Timecode" box. 5DtoRGB will embed a timecode track based on this value and the selected output frame rate. This feature only works with QuickTime output.

Command Line / Shell Scripting Support

5DtoRGB supports basic command line operation when running in CLI (command line interface) mode. Using a terminal, you can pass 5DtoRGB five arguments:

1. An input file path
2. An integer representing the desired output format
3. An integer representing the desired output container (MOV or AVI)
4. An integer representing the decoding matrix value
5. An integer representing the gamma flagging value
6. A float representing the frame rate value

Here's an example:

```
C:\>5DtoRGB.exe \Path_to_MOV\MVI_0001.MOV 3 1 1 0 23.976
```

The first argument, "\Path_to_MOV\MVI_0001.MOV," is a path to an input file. The output file will be written to the input file's directory with the same name as the input file and a "_1.mov" suffix.

The second argument, "3," corresponds to the zero-based index of the desired format in the list of output formats. So, if you want to output the file as DNxHD at 115 Mb/sec, this number would be 3. The supported output formats are as follows:

0. Uncompressed 8 bit 4:2:2
1. Uncompressed 10 bit 4:2:2
2. Avid DNxHD 175 Mb/sec
3. Avid DNxHD 115 Mb/sec
4. Avid DNxHD 36 Mb/sec
5. QuickTime RLE/Animation
6. HuffYUV (RGB)

The next two arguments correspond to the list index of the desired decoding matrix and gamma flagging value, respectively. Like the output format list, this index is zero-based so "0" is the first item in the list.

The last argument is the desired frame rate to use for the transcoded file. This is used to retime shots that have been shot at higher frame rates with "overcranking," or slow motion in mind.

Usage Tips

5DtoRGB will perform at its best when reading from and writing to fast storage media. The faster the storage device, the faster 5DtoRGB can perform read/write operations. A SAN or locally attached disk array is preferable.

It is also possible to do batch conversions using a batch script that utilizes 5DtoRGB's command line interface. Third-party scripts are available for download that support batch conversions.

Requirements

5DtoRGB requires OpenGL support by the video card vendor and operating system. Most modern video cards support OpenGL.

Reporting Bugs

To report bugs or for general feedback, please contact us at support@rarevision.com.

Notes on LGPL Compliance

This software links with software libraries from the FFmpeg project under the terms of the GNU LGPL version 2.1. Source code for these libraries can be found at <http://www.ffmpeg.org>. Full text of the GNU LGPL 2.1 can be found at <http://www.gnu.org/licenses/lgpl-2.1.txt>.