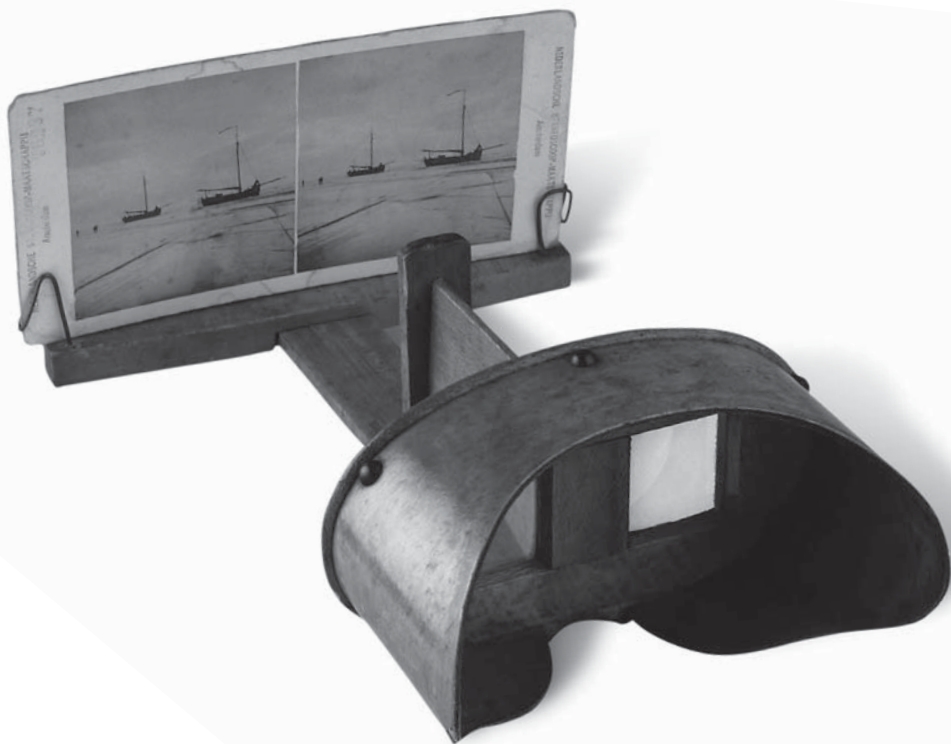


LINUX MULTIMEDIA HACKS™

*Tips & Tools for Taming Images,
Audio, and Video*



O'REILLY®

Kyle Rankin

HACK
#54

Watch Videos in ASCII Art

Use MPlayer's AALib support to convert any movie into ASCII art.

If you have been on the Internet for any amount of time, chances are that you have come across ASCII art (drawings made with the ASCII text characters) either in an email signature, a web site, or some other place. Good ASCII art can take time and talent to look just right, but you can skip through that effort with AALib (<http://aa-project.sourceforge.net/aalib>), a library devoted to converting any image into an ASCII art equivalent. Since a movie is basically a system of moving images, MPlayer has added support for AALib as a video output option. This means that each frame in the movie is converted to an ASCII equivalent and displayed on the screen. This hack describes the basic options needed to convert a video into ASCII art.

To MPlayer, AALib is yet another output format that it can support. Like with other output options, if you have compiled *mplayer* yourself, you will need to compile in support for AALib. Many of the MPlayer packages out there already support AALib, so if you use one of those, you should be fine. To turn on AALib output, just add `-vo aa` to your *mplayer* command:

```
$ mplayer -vo aa video.avi
```

If you run this command from the console, MPlayer will display the movie with your console font and at your console's resolution. If you run this command from a terminal in X, by default a new X terminal will appear and display the movie with a rather large font for each pixel.

The default can be fine just to give a quick demonstration to your friends, but if you want to get higher resolutions out of AALib, you will have to pass it some options. The first set of options to pass are width and height options. Note that these specify width and height in characters, not pixels, so don't simply pass along the resolution of your display. The width and height to use varies based on the size you want your screen and the size of the font you use. There isn't a hard and fast rule, so experiment with some options until you get the results you want. In this example, I will run *mplayer* with AALib set to a width of 250 characters and a height of 80 characters:

```
$ mplayer -vo aa:width=250:height=80 video.avi
```

Even with the width and height settings, chances are that your video will still seem rather pixellated. The AALib output option supports a font argument; however, all of the fonts to choose from are 8-point or larger. To pick

a smaller font, run *mplayer* within a terminal in *ncurses* mode. A terminal has a lot more font choices including some below eight points. A small but common font to use is aliased as 5×7, so start a new *xterm* with that font size with an *mplayer* session executing inside:

```
$ xterm -fn 5x7 -geometry 250x80 -e "mplayer -vo aa:driver=curses video.avi"
```

The resulting video will run inside of the *xterm*; however, unlike with the X11 AALib window, resizing this window will not result in a larger image—you will need to experiment with geometry settings to do that. Also, *ncurses* output is much more taxing on a CPU than the X11 output and even moreso as you increase the resolution, so keep that in mind on a slow CPU.



I've found a very small 2-point font that may or may not be available on your system. If it is, it makes an excellent AALib candidate because it is incredibly small, so you can get even more detail from your video. To use it, type

```
$ xterm -fn "-misc-nil-medium-r-normal--2-20-75-75-  
c-10-misc-fontspecific" -geometry 450x150 -e \  
"mplayer -vo aa:driver=curses video.avi"
```

In Living Color

So you may like AALib, but sometimes those shades of grey just don't cut it. For that, I give you *caca*. *Libcaca* (<http://sam.zoy.org/libcaca>) is much like AALib, except that it goes a step further and attempts to match colors, not just shapes or shades of grey in an image. As a result your videos look even more like the original.

MPlayer supports *Libcaca* as a video output option as well, although you pass extra arguments to it somewhat differently. A basic example of *mplayer* with *Libcaca* is:

```
$ mplayer -vo caca video.avi
```

If you run within X, it will default to an X11 output. To change to *ncurses*, you need to set the *CACA_DRIVER* environment variable, so to launch this within our *xterm*, type:

```
$ xterm -fn 5x7 -geometry 250x80 -e "CACA_DRIVER=ncurses mplayer -vo caca  
video.avi"
```

Keep in mind that *Libcaca* uses even more resources than AALib does, so you may need to turn down the resolution settings to get real-time playback.