Transcoder Version 3.0 for Raspberry Pi
now renamed to "rtranscode" to avoid conflicts with the Debian transcode package

User Manual

Illustrated version of the README file

(c) 2015-2017 by Guenter Kreidl
# Table of Contents

I Introduction...................................................................................................................................3  
II Using rtranscode............................................................................................................................5  
   A) Simple mode...............................................................................................................................5  
      1) Using 4 or 5 arguments and any number of options............................................................5  
      2) Getting arguments from a file..............................................................................................6  
      3) Starting with a named channel..............................................................................................6  
      4) Analyze stream uri................................................................................................................6  
      5) Analyze stream uri and start the stream...............................................................................7  
      6) Add channels to a channel database from a text file containig stream URLs....................7  
      7) Getting help.........................................................................................................................7  
   B) Menu mode..................................................................................................................................8  
      1) Main and streaming menu......................................................................................................8  
      2) Recording transcoded streams.............................................................................................9  
      3) Sub-menus............................................................................................................................10  
   C) Program Options.......................................................................................................................11  
      1) Selecting a configuration file...............................................................................................11  
      2) Selecting a channel database..............................................................................................11  
      3) Set audio bitrate...................................................................................................................12  
      4) Set video bitrate...................................................................................................................12  
      5) Set output image size for SD videos......................................................................................13  
      6) Set output image size for HD videos......................................................................................13  
      7) Path for stream URL..............................................................................................................14  
      8) Selecting the audio output format.........................................................................................14  
      9) Pre-run original stream.........................................................................................................14  
     10) Set recording directory........................................................................................................14  
    11) Run in quiet mode................................................................................................................14  
III Channel Databases..................................................................................................................15  
   A) The default database................................................................................................................15  
   B) Adding groups..........................................................................................................................15  
   C) Create your own channel databases......................................................................................16  
IV The Configuration File............................................................................................................17  
   A) Setting default values..............................................................................................................17  
   B) The really heavy stuff...............................................................................................................19  
   C) Adding modes..........................................................................................................................20  
V Create a web interface for kweb.................................................................................................22  
VI Remote Control......................................................................................................................25  
VII Using http-launch Directly....................................................................................................26  
VIII Copyright, License and Source Code....................................................................................27
I Introduction

My transcoder package contains tools to transcode HTTP TS video streams from a local TV server to a lower bitrate (video and audio) h264 encoded HTTP MKV stream (optionally using lower image size as well), suitable for sending across a low bandwidth connection (through your internet connection, for example). You might use it to watch your home TV from your working place or on your smartphone or let your friends share your TV viewing pleasure. You can also use it to record transcoded streams.

I have successfully used it with mumudvb unicast HTTP streams. It should also work with VDR, but I couldn't get it to work with tvheadend, which doesn't seem to be compatible with the souphtpsrc module of gstreamer.

Note: I know that a lot of people are using tvheadend and may be disappointed that my transcoder package doesn't work with it. I believe it is caused by the crappy way which tvheadend uses to register clients (I've seen a lot of “Cookie synflood warnings” in the messages log, caused by tvheadend). The gstreamer souphtpsrc module, which I have to use to access http streams seems to hang during the registration process and never reaches the DVB stream. At the moment there is nothing I can do about it.

This version also has experimental support for UDP (multicast) streams, but I had no chance to test it myself. It might work with HDHOMERUN or similar hardware.

The transcoder uses the GPU of the Raspberry and gstreamer-1.0 modules and requires a Rasbpian Jessie installation. It installs the following components:

1) a slightly extended version of Sebastian Droege's http-launch
2) a binary "rtranscode" which really simplifies working with http-launch (written in Python and compiled with nuitka)
3) a script "bg_rtranscode.sh" to start rtranscode in quiet mode, suitable for use from other applications.
4) a configuration file and an empty "channels.dat" in /usr/local/share/rtranscode

If you have my kweb suite (Minimal Kiosk Browser, omxplayerGUI) installed, an additional binary "create_kpages" will be installed, which can create a local web interface (HTML files) from a channel database.

Run "sudo ./install" for installation.
**Requirements:**

rtranscode needs the whole set of gstreamer-1.0 modules including gst-omx. To make sure you have everything in place, run:
sudo apt-get update
sudo apt-get install gstreamer1.0-libav gstreamer1.0-omx gstreamer1.0-plugins-bad gstreamer1.0-plugins-base gstreamer1.0-plugins-base-apps gstreamer1.0-plugins-good gstreamer1.0-plugins-ugly gstreamer1.0-tools.

To transcode MPEG2 encoded streams you have to buy the MPEG2 License from the Raspberry Pi Foundation.

For best performance you should overclock the GPU. I use
gpu_freq=500
in /boot/config.txt
Using
h264_freq=500
should also work.

The GPU memory should be set to at least 128 MB.

To use all features you will need a Raspberry Pi 2 or 3. On earlier models it will work, at least for SD channels, but you should avoid audio conversion if the CPU runs too high.

Get the latest official kernel and firmware (using apt-get!).

**Limits:**

SD channels can be transcoded to any size up to 720x576.
1080i25 can be transcoded up to 720p25
720p50 streams cannot be transcoded to 720p50 reliably; maximal output size for 720p50 streams is 1024x576@50Hz.

If you have interlace artifacts in your output stream, reduce the image size to half size of the original stream or less.
II Using rtranscode

The utility "rtranscode" can be used in a number of ways, using different modes.

A) Simple mode

In simple mode rtranscode will set up the arguments and then start http-launch, replacing its own process by it. Once the stream is running, entering "CTRL+c" is the only way to stop it.

rtranscode also offers some utility functions, like analyzing a stream and providing the arguments required for a stream. It can also create a channel database from a text file (e.g. an m3u playlist file) containing stream URLs.

1) Using 4 or 5 arguments and any number of options

rtranscode [options] uri videomode audiomode audiopid [vo]

uri = URL of your local TV http server stream (MPEG TS streams only!)

dvemmode must be one of the following:
sd1 for MPEG2 576i25 SD video
sd2 for H264 576i25 SD video
hd1 for H264 720p50 HD video
hd2 for H264 1080i25 HD video

audiomode must be either 'mpeg', 'ac3' or 'aac' (experimental)

audiopid must match a valid audio pid from your TS stream (decimal number or hexadecimal starting with '0x'). The selected audio pid must match the format set in audio mode. If the stream only contains one audio pid, you can also use "-1".

If the optional fifth parameter "vo" (= video only) is added, audio will not be transcoded and the original audio track (selected by audiopid) will be included.

Example:
rtranscode http://localhost:9082/bysid/11110 hd1 mpeg 0x17e8

The transcoding server will be started immediately, if the original stream is available. To stop transcoding, press CTRL+c.
2) Getting arguments from a file

rtranscode [options] -f=file_path
file_path must either be an absolute path or the name of a file in the current
directory. If the path name contains spaces, it must be quoted. The file must be a
text file containing one line of text like this
uri videomode audiomode audiopid [vo]
(arguments like in 1)

Example:
rtranscode [options] -f="/home/pi/ZDF HD.txt"

The transcoding server will be started immediately, if the original stream is
available. To stop transcoding, press CTRL+c.

3) Starting with a named channel

rtranscode [options] -n=channel_name
This requires a channel database. By default it uses
/usr/local/share/rtranscode/channels.dat, but you can set another database file with
the -d option (see below). If the channel name contains spaces, it must be quoted.

Example:
rtranscode [options] -n="ZDF HD"

If the channel name is found in the database, the transcoding server will be started
immediately, if the original stream is available. To stop transcoding, press CTRL+c.

4) Analyze stream uri

rtranscode [options] -g=URI
where URI is the URL of your original stream.

rtranscode will analyze the output of "omxplayer -i URI" and if successfull, will print
the results like this:
channel_name=uri videomode audiomode audiopid
suitable for inclusion into a database file.

Example:
rtranscode -g=http://localhost:9082/bysid/11110
Result:
ZDF HD=http://localhost:9082/bysid/11110 hd1 mpeg 0x17e8
Note: You can use a redirection to append the result to a channel database:
`rtranscode -g=http://localhost:9082/bysid/11110 >> /home/pi/mychannels.dat`

5) Analyze stream uri and start the stream

```
rtranscode [options] -t=URI
```

where URI is the URL of your original stream.

rtranscode will analyze the output of "omxplayer -i URI" and if successfull, will print the results like this:

```
channel_name=uri videomode audiomode audiopid
```
suitable for inclusion into a database file. It will also start the transcoding server immediately. To stop transcoding, press CTRL+c.

6) Add channels to a channel database from a text file containig stream URLs

```
rtranscode [options] -i=infile -o=outfile
```

infile must be the path to a text file containing stream links. This can also be an m3u file with stream links. infile must be either a complete path starting with '/' or a file in your current directory. If the path contains spaces, it must be quoted.

outfile must be the path to a channel database file. If it doesn't exist, it will be created. outfile must be either a complete path starting with '/' or a file in your current directory. If the path contains spaces, it must be quoted.

rtranscode will try to analyse any stream URL found in infile and add the channel to outfile, if successfull.

Example
```
rtranscode -i=playlist.m3u -o=mychannels.dat
```

7) Getting help

```
rtranscode -h
```
or
```
rtranscode --help
```

will print a simple usage page, showing all different modes and option.
B) Menu mode

1) Main and streaming menu

rtranscode [options]

If started without arguments, rtranscode will run in menu mode. This requires a working channel database. By default it uses
/usr/local/share/rtranscode/channels.dat, but you can set another database file with the -d option, e.g.
rtranscode -d=/home/pi/mychannels.dat

rtranscode uses a curses menu and falls back to a simple menu, if certain problems occur (e.g. too many channels in the database without using grouping).

Note: executing commands always requires a text entry followed by hitting the Return or Enter key.

Here is an example of the menu as you might see it in your terminal:

```
ABR: 64K (a)  VBR: medium (v)  SD-Size: 360x288 (s)  HD-Size: 768x432 (h)
Group: ARD ZDF HD (g) Channels:
Das Erste HD (0)  ZDF HD (1)  zdf_neo HD (2)
ZDFinfo HD (3)  3sat HD (4)  tagesschau24 HD (5)
ONE HD (6)  KiKA HD (7)  SWR BW HD (8)
SWR RP HD (9)  NDR FS NDS HD (10)  NDR FS HH HD (11)
NDR FS SH HD (12)  NDR FS MV HD (13)  hr-fernen HD (14)
MDR S-Anhalt HD (15)  MDR Sachsen HD (16)  MDR Thueringen HD (17)
rbb Berlin HD (18)  rbb Brandenburg HD (19)  BR Sued HD (20)
BR Nord HD (21)  PHOENIX HD (22)  arte HD (23)
WDR HD Koeln (24)
Enter a channel number, 'a','v','s','h', 'g' or 'q' to quit:
```

The first line shows the active settings:

ABR = current audio bit rate for the output stream
enter 'a' and hit "Return" to select another one from a separate menu page.

VBR = current video bit rate for the output stream
enter 'v' and hit "Return" to select another one from a separate menu page.

SD-Size = current output size for SD channels (video modes sd1 and sd2).
enter 's' and hit "Return" to select another one from a separate menu page.

HD-Size = current output size for HD channels (video modes hd1 and hd2).
enter 'h' and hit "Return" to select another one from a separate menu page.
The "Group:" entry in the second line will only be shown, if you use grouping in your channel database. Enter 'g' and hit "Return" to select another group of channels from a separate menu page.

To quit the program enter 'q' in the main menu and hit "Return".

To start transcoding a channel, enter the channel number and hit "Return". The stream menu will be shown. It might look like this:

```
Streaming ZDF HD to http://127.0.0.1:9080/xyz.mkv
Size: 910x512  VBR: 1.85M  ABR: 64K AAC
Recording time: Unlimited

Commands:
's' = stop  'x' = restart transcoding
'r' = record stream  't' = set recording time

Enter command and hit Return:
```

If the stream blocks for some reason, you can enter 'x' and hit "Return" to restart it with the same settings. Entering 's' will stop the stream and return to the main menu.

2) Recording transcoded streams

It's also possible to record the transcoded stream by entering the "r" command. If you want to record for a specific duration, you can set this using the 't' command (before you start recording), which will open a submenu:

```
Current recording time is set to: 0 = unlimited

You can set the new recording time in seconds
or as 'h:m', e. g. '1:30'
or set it to '0' for unlimited recording

Enter new recording time:
```

If you have started recording, the streaming menu will change and may look like this:

```
Streaming ZDF HD to http://127.0.0.1:9080/xyz.mkv
Size: 910x512  VBR: 1.85M  ABR: 64K AAC

Enter 's' to stop recording:
```

You can stop the recording at any time by entering 's' and hitting Return.
Note: If you have set up a recording time, the menu will not change when the recording stops. By entering an empty command (just hit Return) you can check, if the program is still recording. If not, it will show the stream menu again.

3) Sub-menus

Groups menu (example):

Current Group = ARD ZDF HD

Available Groups:
ARD ZDF HD (0)  ARD ZDF SD (1)  News (2)  
Privat TV DE (3)  Sport (4)

Enter a new group number and hit 'Enter':

Audio bit rates:

Current Audio-Bitrate = 64K

Available Bitrates:
original (0) 32K (1) 64K (2) 96K (3) 128K (4) 160K (5) 192K (6) 256K (7) 320K (8) 384K (9) 448K (10) 512K (11) 640K (12)

Enter a new bitrate number and hit 'Enter':

video bit rates:

Current Video-Bitrate = medium

Available Bitrates:
low (0) medium (1) high (2) 192K (3) 288K (4) 384K (5) 448K (6) 512K (7) 640K (8) 768K (9) 896K (10) 1M (11) 1.1M (12) 1.25M (13) 1.5M (14) 1.75M (15) 2M (16) 3M (17) 4M (18) 5M (19) 6M (20) 7M (21) 8M (22)

Enter a new bitrate number and hit 'Enter':

SD video sizes:

Current SD video size = 360x288

Available SD video sizes:
120x96 (0) 180x144 (1) 240x192 (2) 360x288 (3) 480x384 (4) 600x480 (5) 720x576 (6)
HD video sizes:

Current HD video size = 768x432

Available HD video sizes:
256x144 (0)
512x288 (1)
768x432 (2)
910x512 (3)
1024x576 (4)
1280x720 (5)

Enter a new HD video size number and hit 'Enter':

C) Program Options

1) Selecting a configuration file

-c=config_file

By default rtranscode will use '/usr/local/share/rtranscode/config.py' for configuration. You can create your own configuration files and use this option to use one of them instead of the default configuration file.

Note:
config_file must be a valid python file, which will be executed, when the program starts. See chapter IV for details if you want to create your own configuration file(s).

config_file must be either a complete file path or simply a file name. In the second case it will be first searched in '/usr/local/share/rtranscode' and then in your current working directory. If found, rtranscode will try to use it instead of the default configuration file.

2) Selecting a channel database

-d=channels_file

By default rtranscode uses '/usr/local/share/rtranscode/channels.dat' for menu mode or when used with the '-n=channel_name' command line option. This file is empty at installation and you have to add channels by editing it.

You may also create your own channel database files and use the '-d' option to select them.

channels_file must be either a complete file path or simply a file name. In the second case it will be first searched in '/usr/local/share/rtranscode' and then in your
current working directory. If found, rtranscode will try to use it instead of the default channel database.

See chapter III for more details about channel database files.

3) **Set audio bitrate**

-a=audio bitrate number
sets the audio bit rate for the stream or the default audio bit rate in menu mode
number = 0..12, default = 2 (64K)

Audio Bitrates:
0 use original stream
1 32K  32768
2 64K  65536
3 96K  98304
4 128K 131072
5 160K 163840
6 192K 196608
7 256K 262144
8 320K 327680
9 384K 393216
10 448K 458752
11 512K 524288
12 640K 655360

4) **Set video bitrate**

-v=video bitrate number
sets the video bit rate for the stream or the default video bit rate in menu mode
number = 0..22, default = 1 (calculate medium quality)

0 low 12 (divider)
1 medium 9 (divider)
2 high 6 (divider)
3 192K 196608
4 288K 294912
5 384K 393216
6 448K 458752
7 512K 524288
8 640K 655360
<table>
<thead>
<tr>
<th></th>
<th>SD video size number</th>
<th>HD video size number</th>
</tr>
</thead>
<tbody>
<tr>
<td>768K</td>
<td>917504</td>
<td>1.25M</td>
</tr>
<tr>
<td>1M</td>
<td>1048576</td>
<td>1.5M</td>
</tr>
<tr>
<td>1.1M</td>
<td>1153432</td>
<td>1.75M</td>
</tr>
<tr>
<td>1.25M</td>
<td>1310720</td>
<td>2M</td>
</tr>
<tr>
<td>120x96</td>
<td>917504</td>
<td>360x288</td>
</tr>
<tr>
<td>180x144</td>
<td>1048576</td>
<td>480x384</td>
</tr>
<tr>
<td>240x192</td>
<td>1153432</td>
<td>600x480</td>
</tr>
<tr>
<td>360x288</td>
<td>1310720</td>
<td>720x576</td>
</tr>
<tr>
<td>480x384</td>
<td>1572864</td>
<td>7340032</td>
</tr>
</tbody>
</table>

5) **Set output image size for SD videos**

- **s**=SD video size number
  
  Sets the output image size for SD video (modes sd1 and sd2)

  number = 0..6, default = 3 (360x288)

  **SD Video Sizes:**
  - 0: 120x96
  - 1: 180x144
  - 2: 240x192
  - 3: 360x288
  - 4: 480x384
  - 5: 600x480
  - 6: 720x576

6) **Set output image size for HD videos**

- **h**=HD video size number

  Sets the output image size for HD video (modes hd1 and hd2)

  number = 0..5, default = 2 (768x432)

  **HD Video Sizes:**
  - 0: 256x144
  - 1: 512x288
  - 2: 768x432
  - 3: 910x512
  - 4: 1024x576
  - 5: 1280x720
7) Path for stream URL

-p=path

Sets output path for stream URL, must start with a '/' and end with '.mkv', default = "/xyz.mkv"

8) Selecting the audio output format

-u=audio_output_format

allowed values are 'aac', 'ac3' or 'both'
If audio transcoding is enabled, transcode to this format. If using 'both', AAC will be used for MPEG and AAC audio input, AC3 for for AC3 input.

If 'ac3' is selected, the stream analyzer will prefer AC3 channels to MPEG audio channels.

9) Pre-run original stream

-r=delay
delay = 0..10, default = 0

If not zero, access the original stream for 'delay' seconds to make up for tuning time. This will be used for streaming and analyzing.

10) Set recording directory

-l=record_path
Must be a full path to an existing directory, default = current directory

11) Run in quiet mode

-q
Both rtranscode and http-launch will not print anything to the command line. Cannot be used for menu mode. Useful if you call rtranscode from other applications and want to run it in the background.
III Channel Databases

A) The default database

For menu mode and for named streams (-n=name) you need a channel database. By default '/usr/local/share/rtranscode/channels.dat' will be used. This file is created during installation and does not contain anything yet.

Channel database files are simple text files, which are easy to create and edit. To use the default channel database, you have to add some content first (as root). Using nano:

```
sudo nano /usr/local/share/transcode/channels.dat
```

If you prefer a desktop program:

```
gksudo leafpad /usr/local/share/transcode/channels.dat
```

Enter lines of the following form:

channel name=uri videomode audiomode audiopid [vo]

for example:

```
ZDF HD=http://192.168.0.34:9082/bysid/11110 hd1 mpeg 6120
```

(adding "vo" at the end, if you never want to transcode the audio track).

You can use rtranscode's analyze function to get the content of the lines:

```
rtranscode -g=uri
```

but perhaps you want to use another audio channel. To see all channels, run omxplayer -i uri.

The channel names should only contain standard ASCII characters (or the curses menu may break).

Empty lines or lines starting with a '#' (comments) are ignored.

After saving the file it can be used and the channels will appear in the main menu.

B) Adding groups

To make the menu more comfortable to use, you can group your channels. This is really required if your database file contains more than 60 channels (because the curses menu will break otherwise and will be replaced by a simple menu which might require scrolling).

To add a group, add a line like this:

```
[group name]
```

All channel entries following this line, belong to this group.
To switch to another group, add another group line with a different name. You can use the same group multiple times, which means that you can switch back to a group already used before.

If there is no group entry before the first channel line, all following channels will be added to the "default" group, until a new group declaration is is found.

If you have more than one group in your database, the "Group:" entry will be shown in the menu and you can switch between groups by entering 'g'.

C) Create your own channel databases

Instead of using the default database, you can create your own database(s) and select them using the "-d=database" option.

You can create and edit a text file manually or use rtranscode to help set it up, as shown in chapters II.A.5 and 6.

I want to give a simple example how this can be used with mumudvb. I have started mumudvb with a config file to stream the whole Astra S19.2E transponder on frequency 12188 MHz (German TV channels of the RTL group). mumudvb's web interface runs on port 9082. I download the channel m3u playlist with the following command:

```
wget -O rtl.m3u http://localhost:9082/playlist.m3u
```

Now I use rtranscode's database create function:

```
rtranscode -i=rtl.m3u -o=rtl.dat
```

The program will show the streams it is analyzing and finish with the message:
added 12 entries to /home/pi/rtl.dat

Now I can use it for rtranscode in menu mode:
```
rtranscode -d=rtl.dat
```

The menu will show the following channels for transcoding:

```
  RTL Television (0)  RTL Regional NRW (1)  RTL HB NDS (2)
  RTL FS (3)  RTL2 (4)  TOGGO plus (5)
SUPER RTL (6)  VOX (7)  RTL NITRO (8)
RTLplus (9)  n-tv (10)  RTL HH SH (11)
```
IV The Configuration File

rtranscode (and also create_kpages) use a configuration file to overwrite all global settings. The default configuration file is '/usr/local/share/rtranscode/config.py'. You can edit it (as root!) or create a copy of this file and modify this copy.

Note: The configuration file is a Python file which will be executed by the main programs. If you break the Python syntax, it will not work any more and it may even crash the main programs. Therefore I recommend to create your own config file in the following way:

Create a copy in your user directory, e. g.
cp /usr/local/share/rtranscode/config.py /home/pi/myconfig.py

Open this file with Idle for editing. After you have finished editing it, run "Check module" from Idle's "Run" menu. This will first save the file and then show the result (nothing) in the Python console window. If the Python syntax has been broken by your editing it will throw an error and show you the line with the error in the program window.

If no errors are reported, you can test your config file
rtranscode -c=/home/pi/myconfig.py ...

If all runs well, cou can make it your default configuration:
sudo cp /home/pi/myconfig.py /usr/local/share/rtranscode/config.py

A) Setting default values

The configuration file consists of two parts: The first part contains some heavy stuff which may deeply influence the way the program works (and also break everything). This part should only be modified by people who really know what they are doing. The second part contains some default settings which are easy to modify and can be changed by anyone. We will start with this part.

rt_port = '9080'

This is the port on which the streaming server runs. It must be a string (included within single quotes). You may not use a value below ‘1024’.

rt_path = '/xyz.mkv'

http-launch will reject any access which doesn't use the right URL, which is created
by the path value. If you make your streams accessible from the internet (using port
forwarding in your router), you will prevent anybody else besides you and your
friends from accessing the stream. The only way to do this is to use a long and
cryptic path, something like
'/rvh6uw87hkj37.mkv'
It must start with a slash and end with ".mkv" (included within single quotes).

rt_video_bitrate = '1'
rt_sd_video_size = '3'
rt_hd_video_size = '2'
rt_audio_bitrate = '2'
These values set the default values for video bit rate, audio bit rate, SD video size
and HD video size. The values are index values (numbers as text within single
quotes) of the dictionaries in part 1. Run "rtranscode -h" to get a list of allowed
values.

Note: setting a non-existant key may even break the program!

rt_ch6limit = '9'
Start value for using 6ch ac3 audio out, by default '9' (384K) and should not be lower
than '6' (192K). This value is an index value (numbers as text within single quotes)
of the audio_bitrates dictionary in part 1. Run "rtranscode -h" to get a list of allowed
values for audio bitrates.

Use 6 channels for AAC output too, if True
rt_aac_6ch = False
This value (Boolean) may be either 'True' or 'False' (without quotes!). If set to 'True',
6 channel output will also be used for AAC output, depending on the value of
ch6limit above.

rt_audiooutput = 'both'
The value may be either 'both', 'aac' or 'ac3' (within single quotes). If set to 'both',
audio output format will be AC3 if the input format is AC3 and AAC otherwise.

rt_delayed_start = False
This value (Boolean) may be either 'True' or 'False' (without quotes!). If set to "True",
the streaming pipeline will start when the first client connects, otherwise it will
start immediately (default),

rt_prerun = 0
The value must a number between 0 and 10 (no quotes!). If greater than 0, the
stream will be accessed by a dummy module for n seconds (n = value), before
starting the transcoder or the stream analyzer. This may be helpful, if the original
stream is not immediately accessible, because the application needs some time to
tune the channel.

```
rt_channels_per_line = 3
The value must a number greater 0 (no quotes!). It defines how many channels per
line are shown in the menu. Values between 1 and 6 may be useful (depending on
the length of the channel names in your database.
```

```
rt_simple_menu = False
This value (Boolean) may be either 'True' or 'False' (without quotes!). If set to "True",
the simple menu (not the curses menu) will be used.
```

```
rt_stop_delay = 0.2
rt_start_delay = 0.5
These values (floating point without quotes) are used internally and should be left
unchanged.
```

```
rt_channels_dat = '/usr/local/share/rtranscode/channels.dat'
If you want to use another channel database file by default, enter its full path here,
included within single quotes.
```

```
rt_record_path = "
By default this is empty, which means that all recordings go to the current working
directory. You can set a full path to an existing directory instead, e. g. on a hard disk
or a network share.
```

**B) The really heavy stuff**

The data structures in part 1 of the config file (mostly dictionaries) define how the
gstreamer pipelines for http-launch are built. Changing anything here requires a
deeper understanding of both Python data structs and of building gstreamer
pipelines. I won't go into too many details here, just give some hints and examples
for experiments.

The pipelines are built from two templates, rt_av_template (for both video and
audio conversion) and rt_v_template (video conversion only, using one of the
original audio streams). Both templates include a lot of variables, included between
"$" characters, which are replaced by the program at run time, depending on the
mode (SD or HD modes), audio and video bit rates and other settings. The
av_template, for example looks like this:
'http-launch $port$ $path$ video/x-matroska $runmode$ $source$ ! tsdemux parse-private-sections=false name=demux demux.audio$apid$ ! queue ! $audioparser$ ! $audiodecoder$ ! audioconvert dithering=0 ! audio/x-raw,channels=$channels$ ! $audioencoder$ bitrate=$abr$ ! matroskamux name=stream streamable=true demux. ! queue ! $videoparser$ ! $videodecoder$ ! omxh264enc target-bitrate=$vbr$ control-rate=variable ! video/x-h264,stream-format=byte-stream,profile=high,width=$width$,height=$height$,framerate=$framerate$/1 ! h264parse ! queue ! stream.'

The building of the real pipeline depends on the following structures: rt_sources, rt_sd_modes, rt_hd_modes, rt_audio_decoders, rt_aac_encoder, rt_ac3_encoder, rt_video_bitrates, rt_audio_bitrates, rt_sd_video_sizes and rt_hd_video_sizes.

This is the result of two years of experiments and experience with HW accelerated transcoding using gstreamer on the Raspberry Pi. Changing something will often result in disaster.

C) Adding modes

rtranscode by default supports SD and HD modes used by most European standards. If you have access to NTSC based SD streams, for example, you will have to add SD mode(s) and matching resolutions. I will show one example here. We will add a mode for an MPEG encoded video stream using 720x480i30 as "sd3". The sd_modes structure with the new mode added would look like this:

```
rt_sd_modes = {'sd1':
    {'videoparser':'mpegvideoparse','videodecoder':'omxmpeg2videodec','framerate':'25','check':[
        'mpeg2video','720x576','25 fps']},
    'sd2':
    {'videoparser':'h264parse','videodecoder':'omxh264dec','framerate':'25','check':[
        'h264','720x576','25 fps']},
    'sd3':
    {'videoparser':'mpegvideoparse','videodecoder':'omxmpeg2videodec','framerate':'30','check':[
        'mpeg2video','720x480','30 fps']}
}
```

Every mode must contain a matching video parser and video decoder and the frame rate. The check list contains keywords for the analyze function (video codec, image size, fps).

This mode also requires different SD video sizes to get an undistorted image in the player. We will add them to rt_sd_video_sizes starting with key '7' like this:
rt_sd_video_sizes =
{
'0':"120x96", '1':"180x144", '2':"240x192", '3':"360x288", '4':"480x384", '5':"600x480", '6':"720x576",
'7':"144x96", '8':"216x144", '9':"288x192", '10':"360x240", '11':"432x288", '12':"504x336",
'13':"576x384", '14':"648x432", '15': "720x480"
}

Note: The number of lines of the video output size MUST be a multiple of 16!

Now you can use mode "sd3" in your channel database and this mode should also be discovered by the analyze function.
Create a web interface for kweb

rtranscode is a command line program with an optional simple curses menu running in a terminal. If you want a more comfortable desktop application, you can use tools from my kweb suite (Minimal Kiosk Browser, omxplayerGUI) and the program create_kpages to build a web interface running in kweb or kweb3. Both kweb and kweb3 are simple, but fully functional web browsers with a unique feature: they can execute any kind of program from links, buttons or web forms inside a (local) HTML file without having to run a server backend.

create_kpages creates one or more (if you use groups) HTML files from an existing channel database:

create_kpages [options] [channels database]

Without any options and arguments, this will create a channels.html file from the default database /usr/local/share/rtranscode/channels.dat in your current working directory. If your channel database contains groups, a separate page for each channel will also be created and the main page (channels.html) will contain a group menu.

This image shows an example with omxplayerGUI running on top playing the transcoded stream:

To create a web interface for another channel database, just add the file name or
complete path as argument, e. g.
create_kpages [options] mychannels.dat
If you do not use a full path, the file is first searched in /usr/local/share/rtranscode
and afterwards in the current working directory. The resulting (main) HTML file
will always have the name of the channel database file with the extension ".html"
instead of ".dat".

Options:

-c=config file
(default = /usr/local/share/rtranscode/config.py)

-p=path
output path for stream URL, must start with a '/' and end with '.mkv', default = "/xyz.mkv"

-s=style
file name of a css file from /usr/local/share/kweb, default = "about.css"

Now open the main HTML file from kweb(3). If your channel database contains
groups, you will see a button for each group in the top line and the first group will
be shown below. Each group page (or the main page, if you don't use groups)
presents a web form with the following options:
A menu button to select the video bit rate.
A menu button to select the audio bit rate.
A menu button to select the size for SD video.
A menu button to select the size for HD video.
A menu button to select the pre-run time (if required).

Inside a larger, scrollable menu box you will see all the channels from your
database file (or group). Select a channel and click the "Transcode" button to start
transcoding. A terminal will pop up showing nothing (because the transcoder is
running in silent mode). I'll show you later, how to get rid of it, but for testing it is
nice to have. If the terminal closes again after a moment, the source stream is not
available.

If you click the "Play Transcoded Stream" button, omxplayerGUI will open and show
the stream.

To stop transcoding, click the "Stop Transcoding" button.
Note: you can start a new transcoding without clicking on "Stop Transcoding" first (the old instance will be closed automatically).

To get rid of the terminal, open kweb's menu page, click on "Settings" and scroll down to the direct_commands list. Add a new line at the bottom bg_rtranscode.sh and click the save button beside the list.
rtranscode runs in a terminal (command line, lxterminal, SSH connection). If you close the terminal, all programs started from it will be closed and this will also stop transcoding.

If you want to manage transcoding from the internet, you must establish an SSH connection first. You can run the transcoder, but it will be stopped, if you close the SSH connection. This may not be what you want, but there is a simple solution.

To continue the programs started from a terminal or SSH connection even when the connection is closed, you can use an application named “screen”. You may have to install it first:

```
sudo apt-get install screen.
```

Now you can start rtranscode in the following way:
```
screen rtranscode [options]
```

You can use the program as usual. To detach from the program, press “CTRL+a”, followed by 'd'. The screen session is closed and the program will continue to run in the background. You can now safely close the terminal or SSH connection.

To connect to the running program again later on (from a new terminal or SSH connection), enter:
```
screen -r
```

If you stop the program, the screen application will also close.

Note: I plan to add a (Python based) web frontend to control transcoding in the future, which will be available as a separate package.
VII Using http-launch Directly

You may want to experiment with using http-launch directly. It can be called with:
http-launch PORT PATH MIMETYPE RUNMODE <launch line>

PORT = http server port
PATH = URL path, e.g. "/xyz.mkv"
MIMETYPE = the mime type matching your stream
RUNMODE must be one of the following:
default|silent|delayed|silent-delayed
<launch line> must be a gstreamer1.0 tool chain as you can use it with gst-launch-1.0 with two specialties:
The final muxer must be named "stream" and the final argument should be "stream." Here is an example from rtranscode itself:

http-launch 9080 /xyz.mkv video/x-matroska silent souphttpsrc
location="http://192.168.0.34:9082/bysid/28106" is-live=true keep-alive=true do-
timestamp=true retries=10 typefind=true blocksize=16384 ! tsdemux parse-private-
sections=false name=demux demux.audio_0066 ! queue ! mpegaudioparse !
mpg123audiodec ! audioconvert dithering=0 ! audio/x-raw,channels=2 ! avenc_aac
compliance=-2 bitrate=65536 ! matroskamux name=stream streamable=true demux.
! queue ! mpegvideoparse ! omxmpeg2videodec ! omxh264enc target-bitrate=864000
control-rate=variable ! video/x-h264,stream-format=byte-
stream,profile=high,width=720,height=576,framerate=25/1 ! h264parse ! queue !
stream.
VIII Copyright, License and Source Code

http-launch:
Copyright (C) 2013 Sebastian Dröge <slomo@circular-chaos.org>
Modifications by Guenter Kreidl <gkreidl@krefelder-life.de> Copyright (C) 2015-2017

rtranscode and create_kpages:
Copyright (C) 2017 Guenter Kreidl <gkreidl@krefelder-life.de>

Documentation (README and manual):
Copyright (C) 2017 Guenter Kreidl
It may be distributed without charge in digital form, but not modified in any way.

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

These programs are distributed in the hope that they will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

The source codes are part of the distribution. You will find them in the "sources" directory.

rtranscode and create_kpages are written in Python and compiled with nuitka. You can use the compile script to compile modified versions (nuitka required).

http-launch.c contains my modifications to the original program by Sebastian Droegge. You cannot compile it directly.

Clone from https://github.com/sdroege/http-launch up to commit 5e7b8a936bfcb2f86f6e78c69929d5ace617dc7. Replace the file http-launch.c with the version I supplied. Run ".configure" and then "make".