pootlecam Documentation

Release 0.1a1

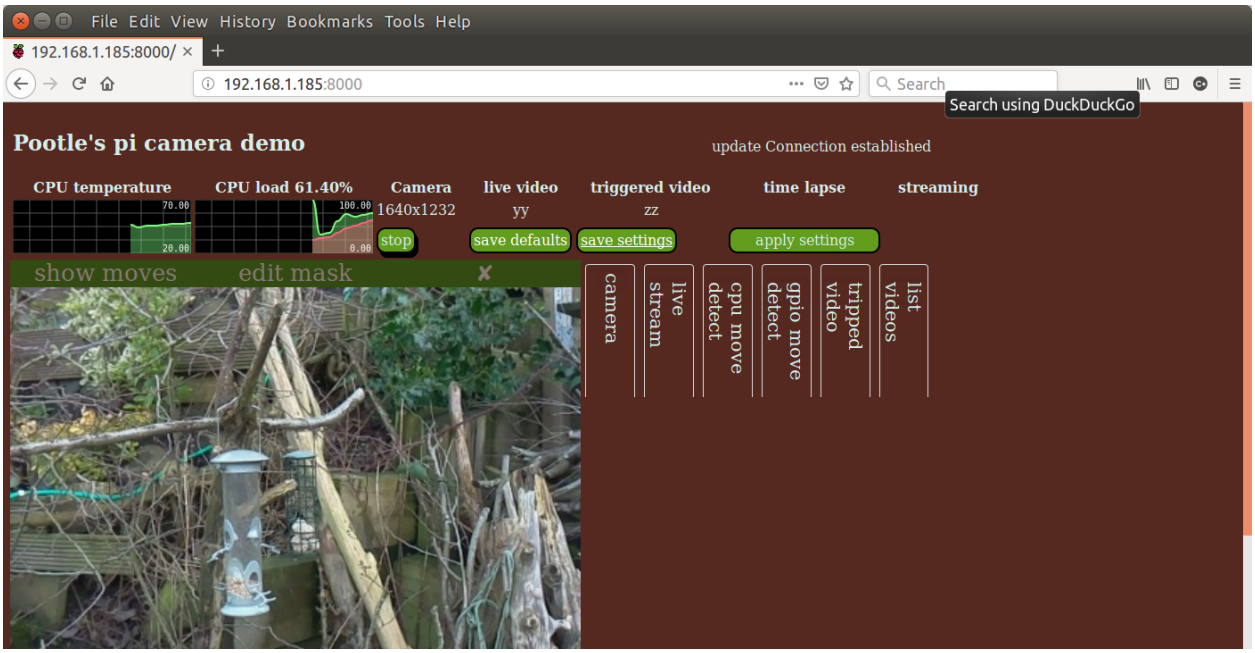
Iain

Feb 11, 2019

Contents:

Pootlecam is a python program to use a Raspberry Pi camera that can do various camera related things such as live streaming, video recording, movement detection and more. It exploits the Raspberry Pi's camera integration with the GPU to enable all this to run well even on a Raspberry Pi Zero. It makes extensive use of the picamera python module, and much of the camera related code is based on examples from that module.

This is the user documentation, it is for anyone that wants to use pootlecam without delving into the python that makes it all work.



CHAPTER 1

Installation and first setup

These notes explain how to get pootlecam running on a clean build of Raspbian Lite (or any other Raspbian build).

Much of the installation can be done with a simple setup script included in the repository.

1.1 Setup using the script

This setup can be used on a fresh build of raspbian, and includes an update of Raspbian at the start of the script. You still need to use raspi-config to set the network name, change the pi user's password and enable the camera.

The time from first power on of a fresh install to complete the script is about 11 minutes on a Raspberry pi Zero, but this will depend on how many updates have to be installed.

If you installed Raspbian lite then just login, if you have the full version of Raspbian, start a shell window.

After logging into your Raspberry pi, first fetch the software from the repository:

```
wget https://github.com/pootle/piCameraWeb/archive/master.zip
```

Then unzip all the files:

```
unzip master.zip
```

and rename the folder (cos its a bit long and clumsy by default):

```
mv piCameraWeb-master piCameraWeb
```

Note that the name piCameraWeb is built into some of the later instructions, so if you use a different folder name some of the things below need adapting.

Now switch to the folder with the setup scrip[t and related files:

```
cd piCameraWeb/setup
```

and finally run the setup script:

```
sudo ./setup.sh
```

1.2 check the camera

To confirm the camera is working, use this command:

```
vcgencmd get_camera
```

this will return 'supported=1' if camera support is enabled and 'detected=1' if a camera is detected.

or you can take a photo by using the command:

```
raspistill -v -o test.jpg
```

After a couple of seconds the command should finish, check for the file (test.jpg) in the current directory. If it is not there, then either the camera is not enabled in raspi-config (the last line of the raspistill output will say this), the camera is not properly connected, or (unlikely) the camera is broken.

1.3 Preparing to run

If you need to use gpio (e.g. PIR motion detection via a GPIO pin) arrange to start the pigpio daemon on boot.

First check where pigpiod is installed:

```
whereis pigpiod
```

Then arrange for it to start on boot:

```
sudo crontab -e
```

and add the line (amend the location if whereis shows it is somewhere else):

```
@reboot                 /usr/bin/pigpiod -c 256 -s 10
```

The -c 256 parameter runs the daemon in realtime mode - that is it is given a high CPU priority.

The -s 10 parameter means that the daemon runs with a sample rate of 10 microseconds, rather than the default of 5 microseconds. This reduces the CPU load especially on a Raspberry Pi Zero. For this application a few extra microseconds to trigger something is not important.

If you want to start pootlecam automatically on boot a simple way is:

```
crontab -e
```

and add this line at the end of the file:

```
@reboot                 ~/piCameraWeb/start.sh
```

(adjust the folder name to match the folder this package is in.)

1.4 Test running

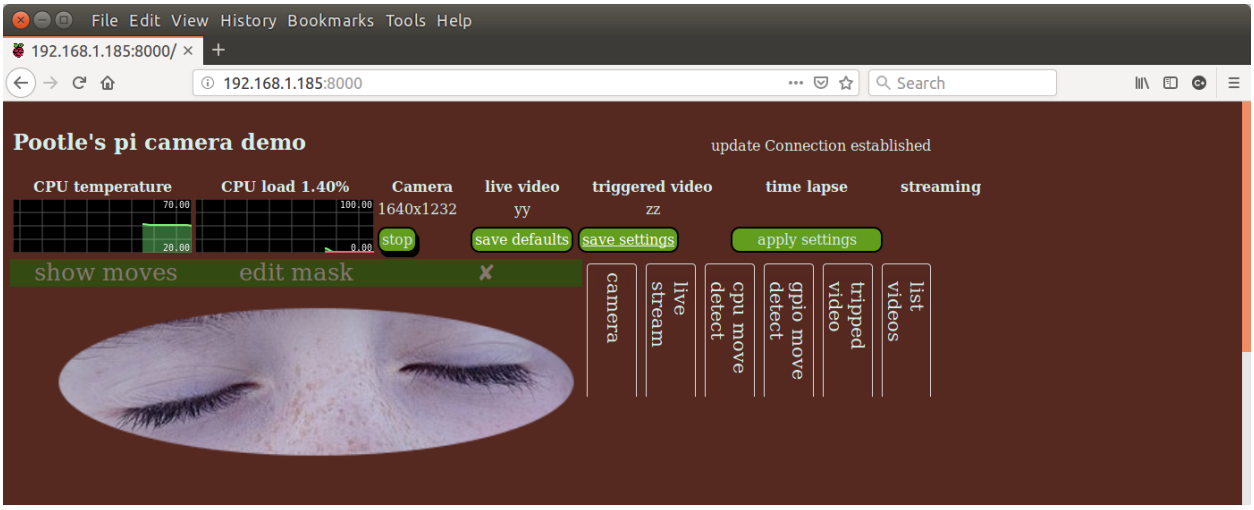
Before rebooting, or if you don't want the start script run automatically, you can run the program interactively. This is a slight variation on the command used in start.sh as it logs messages to the console.

Remember to cd piCameraWeb (or whatever folder you installed to).:

```
python3 webserv.py -v 30 -c dummyConfig.py
```

The first line logged identifies the ip address and port that the web server is using. Copy and paste this to a web browser to see the web site. You can run the web browser on any local PC, phone or tablet. (Local means in network terms, on the same subnet)

You should now see a screen something like this:



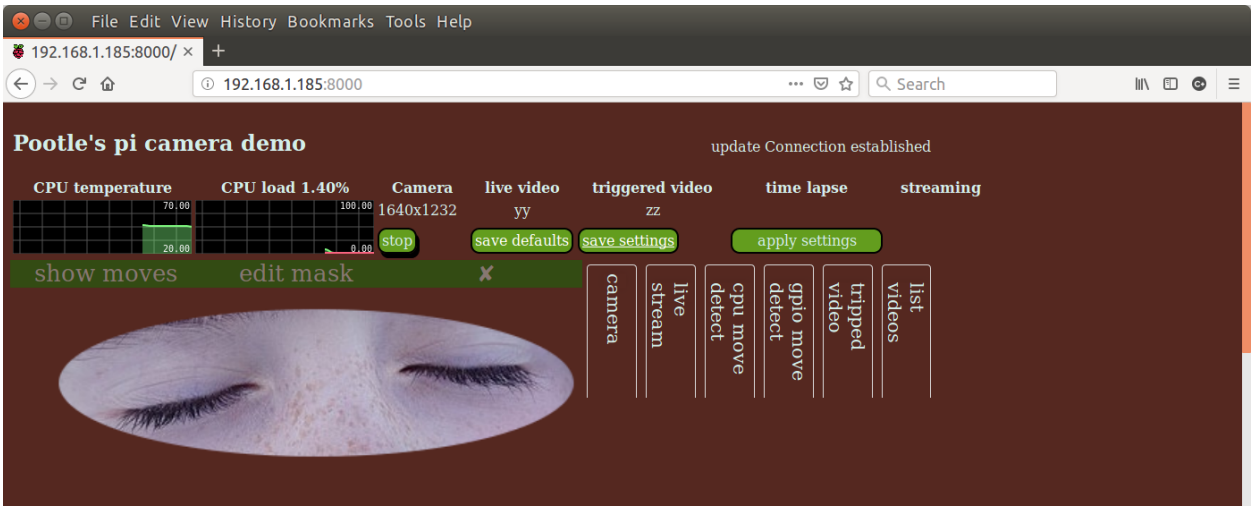
1.5 Power saving

Especially if running from solar or a battery pack, see power saving info here.

CHAPTER 2

The main screen

When you first open the web page, the screen, which will look something like this,



has 3 main areas:

- system summary and control - an overview of the overall system and some top level controls
- live view - when active - a default static image is shown initially
- tabbed panels with detailed controls for the various functions available

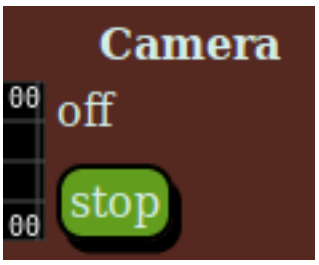
2.1 system summary and control

This area starts with two rolling graphs followed by top level controls:

- the current and recent CPU temperature - useful especially if your pi is in an enclosure
- the overall CPU utilisation - useful to make sure the Raspbery pi isn't getting too busy

- some further controls like the camera control (the rest is still under construction)

2.2 The Camera Status



The camera control includes a simple status field and a button.

The status field typically shows:

- off - if the camera is not running
- the resolution the camera is using if it us running (e.g. 1296x972)

The camera is automatically turned off if nothing is using it after a timeout (the timeout can be set in the camera tab,

The button - "stop" - stops the camera, but only if nothing else is active. Changes to frame rate and overall resolution only take effect when the camera starts. If it is already running, it needs to be stopped and restarted if either of these fields is changed.

Several aspects of the camera's operation can be controlled through the Camera tab see camera tab info here.

2.3 Live View

The live view area is initially inactive. clicking anywhere on the static image will start live view. The live view's controls are in the live stream tab (see tabbed panels below or the detailed description of the live stream tab.

Note that on Raspberry pi Zero the live view uses significant resources - reducing the resolution will reduce this slightly. Also if other activities are running (like triggered video) a Raspberry pi zero does occasionally cause a kernel crash and stops the camera. This doesn't seem to happen on Raspberry pi 2 or 3.

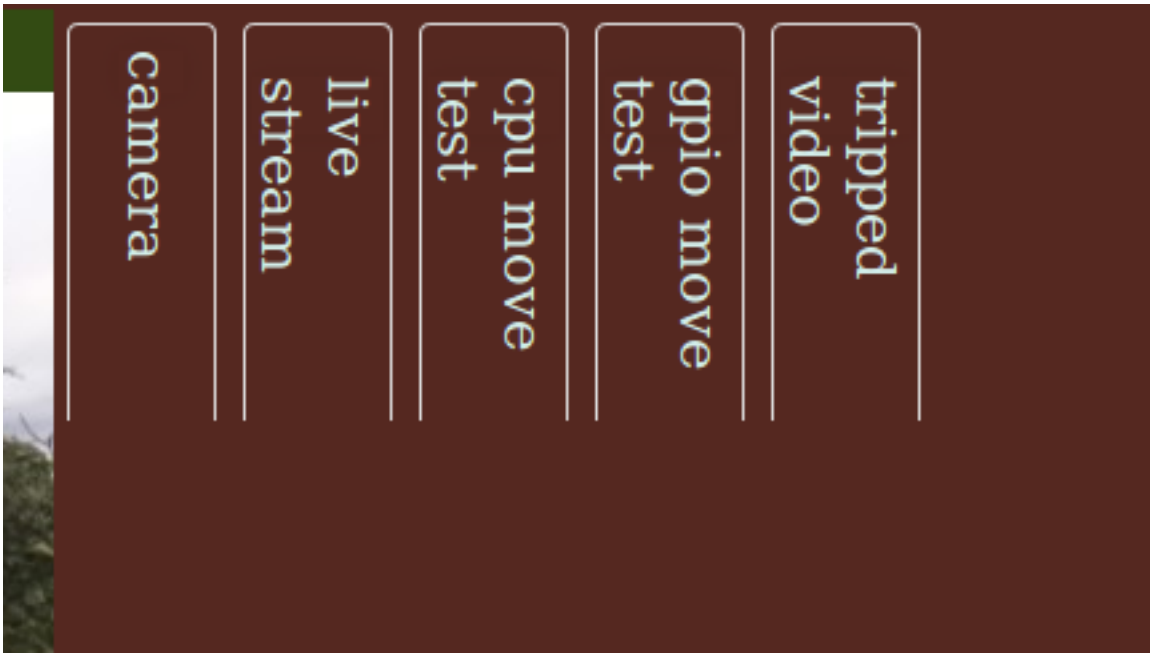
The live view can be stopped at any time by clicking on the 'X' by the top right of the live view area.

Immediately above the live view is the 'edit mask' button that overlays the live view with a mask used by the CPU movement detection activity. This is explained in *mask editing*.

Internally the live view stream is generated once and then replicated to all connected wenb browsers using a live view, this does put significant load on the network. Reducing the frame rate helps a lot.

2.4 Tabbed panels

There are more detailed control panels for the various functions in the tabbed panel area. Initially only the headings are shown, clicking on any heading will highlight the heading and display the relevant controls below the headings.

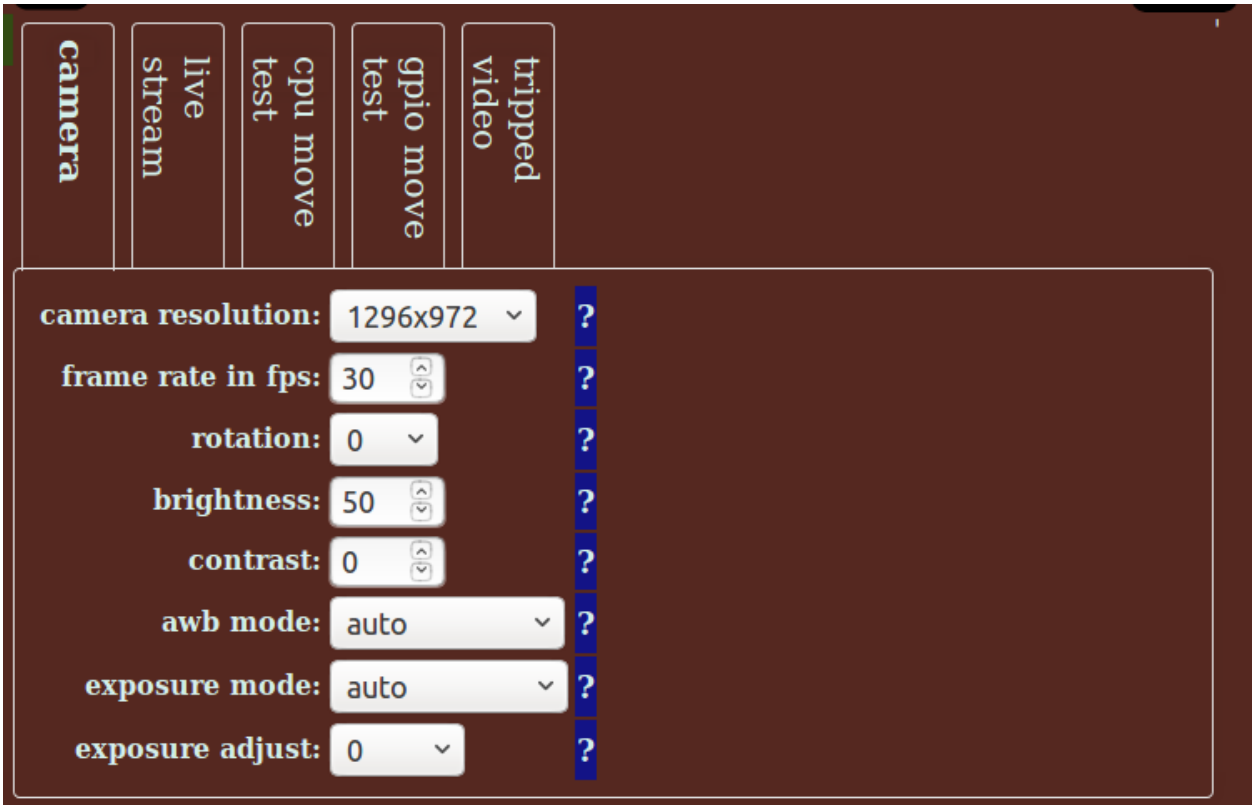


The following pages explain each of the tabs.

CHAPTER 3

Camera tab

The camera tab shows the controls available for the camera.



camera resolution The resolution used when opening the camera. This defaults to 1/2 the size of the camera's full resolution. This reduces the overall load, but still gives the full field of view of the camera. Various other resolutions are available, these vary depending on the camera version. Changes to resolution only take effect if the camera is completed stopped (i.e. before first use or when the camera is restarted)

frame rate in fps This sets the frame rate the camera runs at. Note that this also limits the exposure time, so for viewing in low light, the frame rate must be reduced. This parameter accepts values down to .1, which means 1 frame every 10 seconds. This allows the camera to use an exposure time of up to 10 seconds (for V2 cameras). Note that for longer exposures.'exposure mode' should be set to 'night' As with camera resolution, any changes while the camera is running will not take effect until the camera is restarted.

rotation allows the camera output to be rotated in 90 degree increments to compensate for the camera installation being on its side or upside down. Changes are immediate.

brightness Allows the camera's brightness atribute to be dynamically updated. It usually takes a few frames for any change to stabilise.

contrast Allows the camera's contrast attibute to be dynamically updates. It usually takes a few frames for any change to stabilise.

awb mode Allows the camera's automatic white balance to be dynamically updated. It usually takes a few frames for any change to stabilise.

exposure mode Allows the camera's automatic exposure mode to be dynamically updated. It usually takes a few frames for any change to stabilise.

exposure adjust Provides an exposure compensation adjustment to be dynamically updated. This works in the same way as the camera's attribute and allows adjustment in 1/6th stop increments over a range of + / - 4 stops, although the entire range is not usually effective.

CHAPTER 4

live stream tab

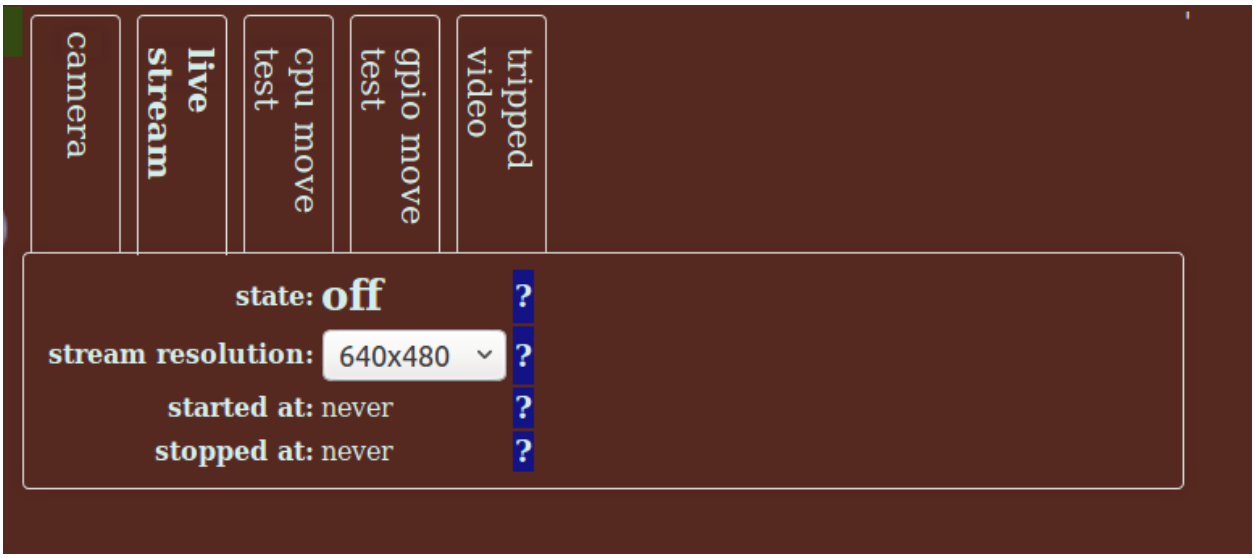
The live stream tab shows the status of the live stream feed and allows the resolution to be set.

The default resolution (640 x 480) provides a reasonably detailed view without loading the system too much.

On Raspberry Pi Zeros, if there are other streams active, it can help to turn the resolution down if you want it permanently on view.

Reducing resolution also helps if there are network problems (such as a poor wifi signal to the raspberry pi).

Note that on the main screen, the live stream is always the same size on screen - even when changed here. The stream is resized in the web browser to do this. This reduces network bandwidth and load on the Raspberry Pi.



4.1 displaying the live stream in other web pages

The live stream can also be viewed from other web pages. The page 'vstream.mjpg' within the website will display the live stream feed. (for example 'http://192.168.0.123:8000/vstream.mjpg'). Viewing this way will show the real size of the stream.

Note that at higher frame rates only a small number of streams can be active before the wifi will start to choke. Lan (Ethernet) connection will improve this somewhat (using a dongle on a raspberry Pi Zero).

4.2 live stream fields

Note that there is no button to start / stop the live stream, as the stream is started and stopped from the live stream area on the main screen.

state Shows the current state of this stream. See *state field for details*.

stream resolution The feed from the camera (which can be of much higher resolution) is scaled to this size. See Common Tab fields for details.

started at The local time the stream last started. See *started at field for details*.

stopped at The local time the stream last stopped. See *stopped at field for details*.

CHAPTER 5

cpu move tab

The cpu move tab shows the settings and status for the cpu based movement detection.

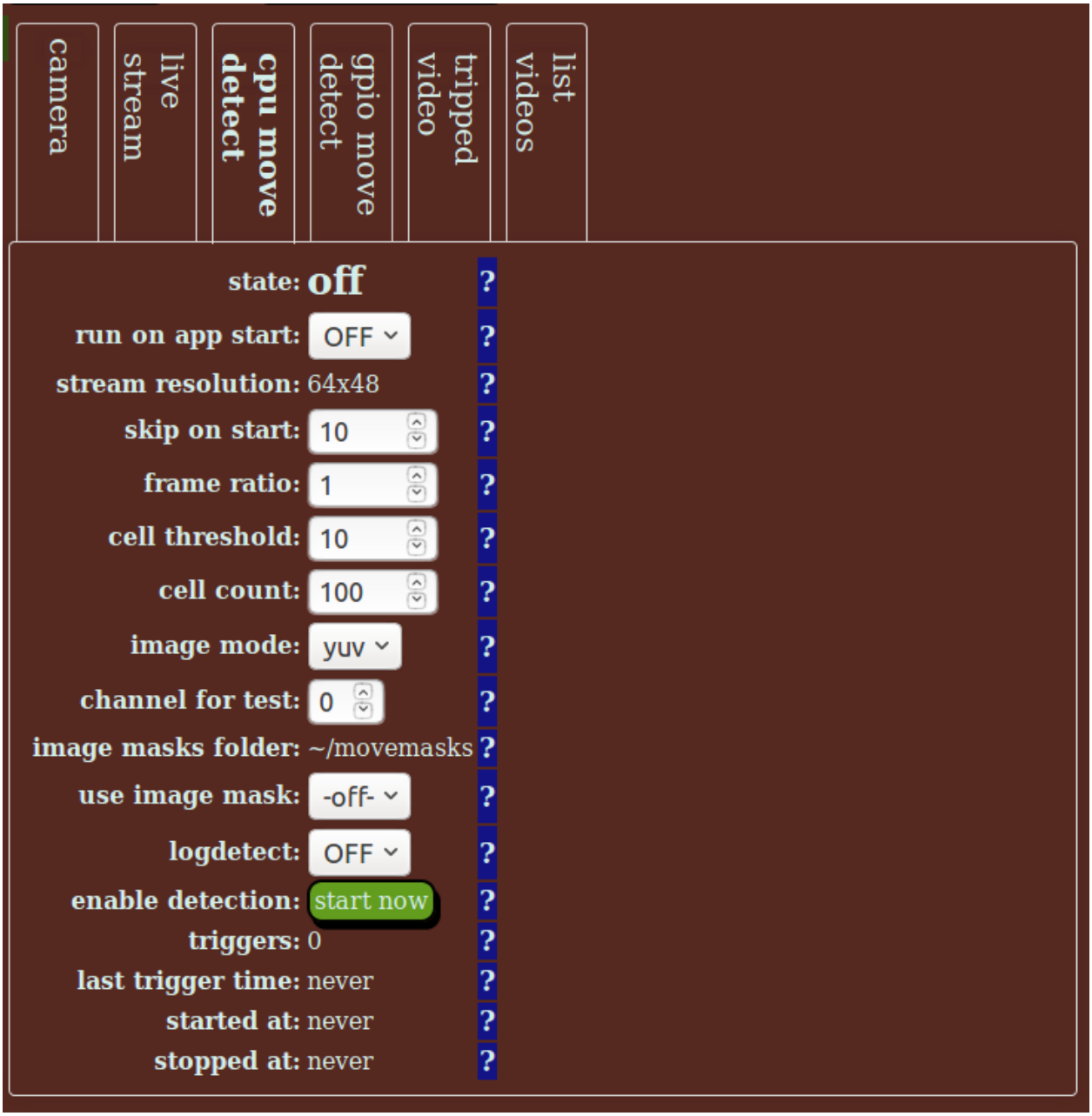
The cpu based movement detection compares successive frames to see how much has changed, and raises a trigger based on the size of change in individual pixels and the number of pixels that reach a threshold value. The trigger is typically used to start a video recording or taking a still image. A mask can be applied to stop particular areas from causing triggers.

The video feed is resized to a currently fixed size of 64 x 48 before analysis.

The overall cpu used to do this is actually quite small even on a raspberry pi zero.

When movement is detected, the ‘triggers’ field is incremented and the ‘last trigger time’ field is set to the current time.

Changes to this field are notified to the main application, which will trigger the ‘triggered video’ stream if it is active.



5.1 Overall detection process

- The main video feed is resized using a pi camera splitter port to the size defined in the field 'stream resolution. This is carried out in the GPU so is fast and efficient. The feed converts each frame to 'rgb' or 'yuv' as defined by the field 'image mode'. 'yuv' mode means it is easy to test just the "brightness" of each pixel to reduce cpu load while still being sensitive.
- if frame ratio is > 1 then the specified number of frames are discarded between each analysis.
- If a mask is active then all pixels that are masked out are set to be ignored.
- The absolute difference between every unmasked pixel in the current and previous images is calculated, using only the channel in the field 'channel for test'.

- pixels are marked if the difference is greater than the field 'cell threshold', otherwise they are unmarked.
- if the count of marked pixels is > the field 'cell count' then the trigger fields are changed as described above.

5.2 cpu move fields

state Shows the current state of this stream. See *state field for details*.

stream resolution The feed from the camera (which can be of much higher resolution) is scaled to this size. See *stream resolution field for details*.

skip on start movement detection skips this number of frames before starting to look for movement. The camera uses a few frames to settle exposure and various other parameters when it first starts. This avoids these frames from raising false triggers

frame ratio If the movement detection code gets to cpu intensive, this reduces the number of times detection actually runs. 1 means every frame is analysed, 3 means every 1/3rd frame is analysed.... Currently the detection code will easily handle 30 frames per second at the set resolution.

cell threshold The difference between the pixel values is tested for each cell and only cells equal to or above the threshold are marked as changed. Note that the fairly small image size used currently also significantly reduces noise in the image to make this less likely to trigger accidentally

cell count When all the cells have been tested for threshold, this is number of cells that must be marked as changed to cause a trigger event.

image mode Either 'rgb' or 'yuv', yuv using channel 0 seems to work well.

channel for test Channel to be used for test, can be 0, 1 or 2. This refers to the the channels in the image, so for 'rgb' they are the red, green or blue channels, and for 'yuv', they are luminance or 1 of the chrominance channels.

image masks folder The folder the mask images are stored in - current fixed in the fields initialisation

use image mask The file in the image masks folder to use as a mask before checking for movement. This can also be 'off' if no mask is to be used.

start now This button starts cpu based movement detection if it is not running, and stops it if it is running.

triggers The count of triggers in the current run of cpu based movement.

last trigger time The last (local) time a movement trigger happened.

started at The local time the stream last started. See *started at field for details*.

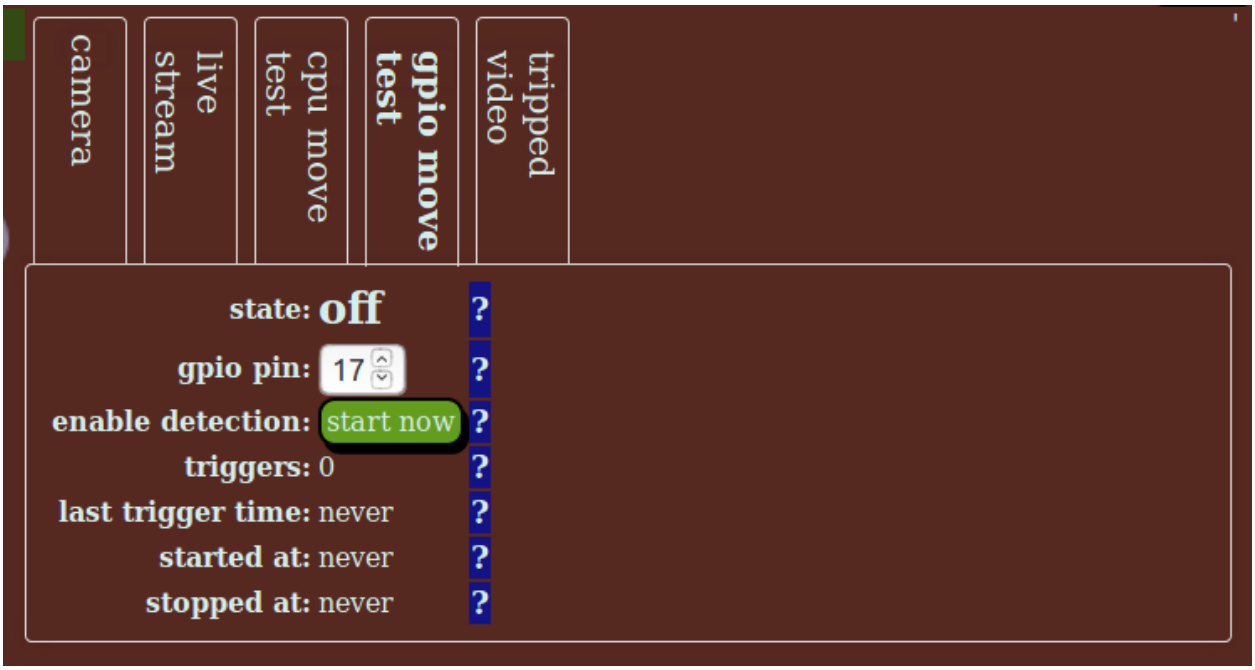
stopped at The local time the stream last stopped. See *stopped at field for details*.

CHAPTER 6

gpio detect

This gpio detect tab allows use of a gpio port as a trigger, typically to trigger a video recording.

This enables a PIR sensor, a button or any other external hardware capable of interfacing to gpio to be used as a trigger.



6.1 gpio move fields

state Shows the current state of this stream. See *state field for details*.

start now This button starts the gpio monitor if it is not running, and stops it if it is running.

triggers The count of triggers in the current run of gpio input detection.

last trigger time The last (local) time a movement trigger happened.

started at The local time the stream last started. See *started at field for details*.

stopped at The local time the stream last stopped. See *stopped at field for details*.

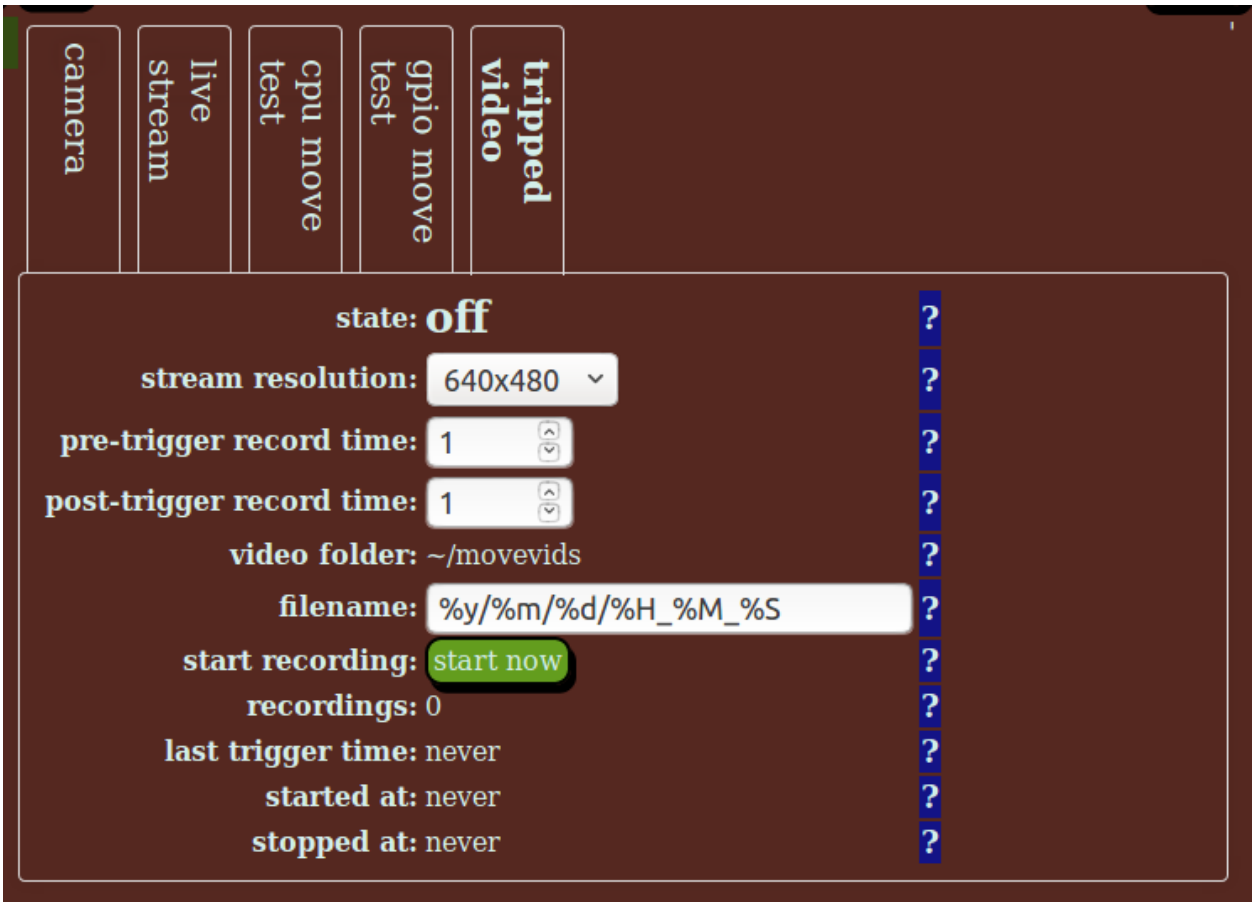
CHAPTER 7

tripped video

The tripped video tab controls and enables video recordings to be triggered by other activities. So far this means cpu based movement detection or using external hardware that can interface with a gpio input pin.

It allows a few seconds of video from before the trigger to be combined with video that runs until several seconds after the last trigger occurs.

It works by continuously recording to a circular buffer, and when triggered, it switches recording to a temporary file, and saves (part of) the circular buffer to another temprary file, and finishes by using MP4Box to merge the 2 files into an mp4 file.



7.1 tripped video fields

state Shows the current state of this stream. See *state field for details*.

stream resolution The feed from the camera (which can be of much higher resolution) is scaled to this size. See *stream resolution field for details*.

pre-trigger record time This is the number of seconds from before the trigger happens to use at the start of the video. The time is approximate and can vary, expecially at low frame rates.

post trigger record time The video keeps running until this number of seconds after the last trigger time.

video folder This field needs changing it doesn't work properly. Should be the base folder for video recordings

filename The filename used for video files. Can include folders as well as the final file name.

start recording This button starts the recorder so it will create video files when triggered. If it is already running, it stops it.

recordings The count of recordings in this session

last trigger time The last (local) time a video was triggered

started at The local time the stream last started. See *started at field for details*.

stopped at The local time the stream last stopped. See *stopped at field for details*.

CHAPTER 8

Common tab fields

There are a number of fields that appear on multiple tabs, this section provides more details on these fields than the individual tab sections.

8.1 state

Most activities have a small number of standard states that they use:

- off - The activity has not run since the web service started
- run - The activity is currently running, the time it started is shown in the 'started at' field
- complete - the activity was running, but has now stopped. The time it stopped is shown in the 'stopped at' field

8.2 stream resolution

Activities that use the camera splitter port can resize the video output of the camera to a more suitable size for their own purposes. As this resizing is handled by the GPU is is faster and more efficient than code running in the CPU, as well as leaving the CPU to do other work.

A list of built in resolutions are provided, mostly 4x3 to match the aspect ratio of the camera sensor. The list is tailored to the model of camera in use (V1 or V2) and the cpu based motion detection has a special list that goes to smaller sizes than that ysed elsewhere.

8.3 started at

This field is one of the last on most tabs. It records the last time the activity started. Before recording a specific time, it will be 'never'

8.4 stopped at

This field is one of the last on most tabs. It records the last time the activity stopped. Before recording a specific time, it will be ‘never’

CHAPTER 9

Mask editing

The cpu movement detection can use a mask to prevent triggering from specific areas of the view.

The mask is edited on the main web page using an overlay on the live view.

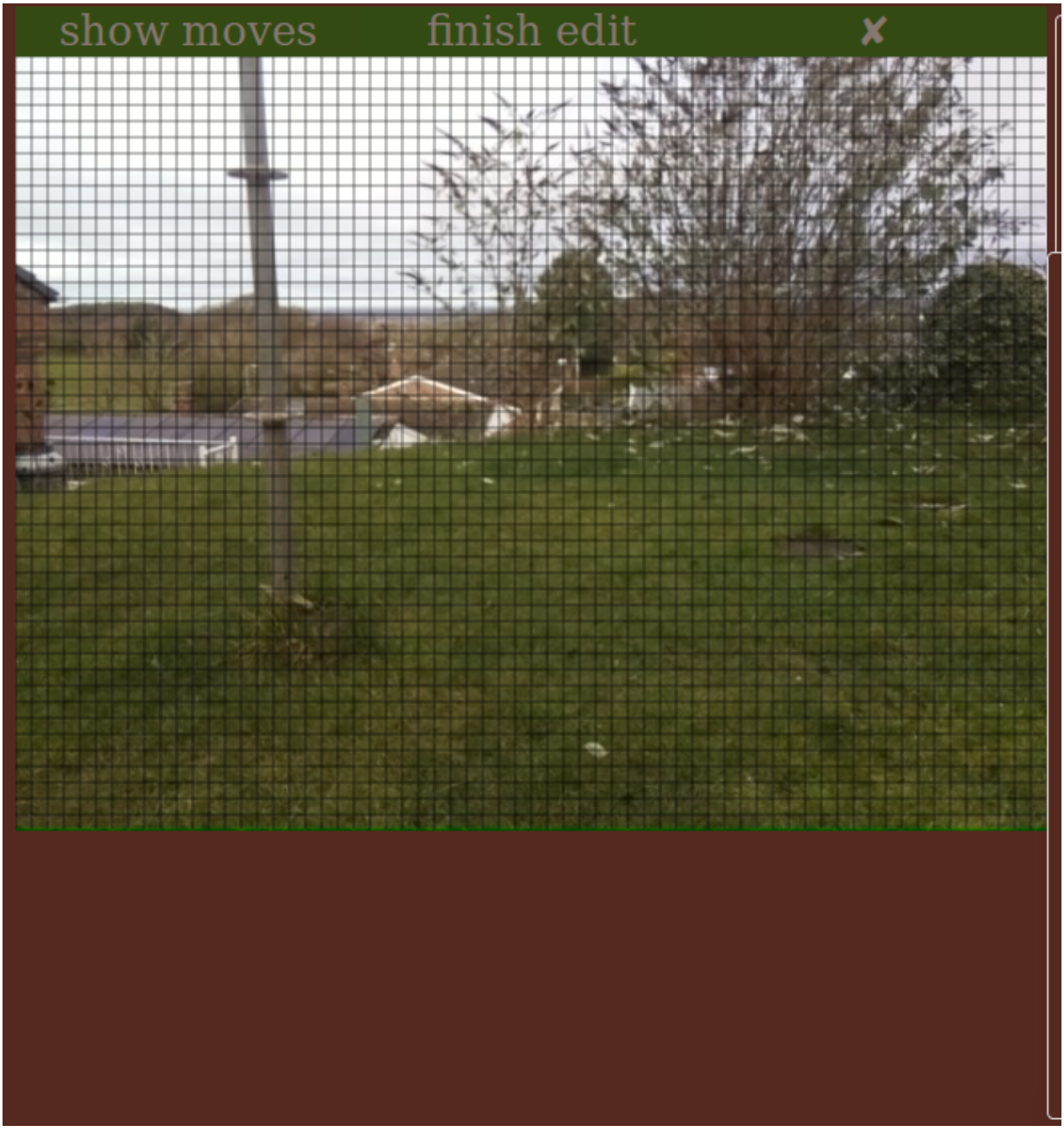
You can either create a new (blank) mask, or edit a previously saved mask. Use the

cpu move tab field: use image mask

to set the mask file. If the mask file is 'off' a blank mask is used, if the mask file is valid the mask file is used as the initial mask to edit.

When the live view is active, clicking on 'edit mask'

overlays the live view with a grid. The overlay shows the individual cells of the mask. 'edit mask' changes to 'finish edit'.



Use the mouse to turn on or off blocks of cells in the mask. Initially all cells are off (NOT masked). Left click on any cell will enable that cell in the mask, and by holding the mouse down and dragging, a rectangular area can be enabled in a single operation. Just release the mouse button to complete an area. Masked areas have a blue wash applied.

To turn off a cell or block of cells hold the shift key down as the mouse button is clicked. (Shift can be released after the mouse button is pressed).

Once editing is complete click on 'finish edit' above the live view. A small prompt screen appears.

Enter a name for the mask in the field provided and the mask is saved on the raspberry pi. Cancel or an empty name discards the edited mask.

Once the mask has been saved, it can be selected on the

cpu move tab field: use image mask

CHAPTER 10

Power savings

Here are some of the ways to save power.

10.1 Turn off hdmi

edit:

```
sudo nano /etc/rc.local
```

and add these lines BEFORE the exit 0:

```
# Disable HDMI
/usr/bin/tvservice -o
```

10.2 Turn off the camera LED

The camera LED is quite bright and in low light can cause reflections as well as making the pi 'obvious' at night.

edit:

```
sudo nano /boot/config.txt
```

and add the following at the end:

```
# Disable camera LED
disable_camera_led=1
```

10.3 Turn off activity LED -Pi ZERO ONLY!

LEDs on other versions require different settings. 'See here <https://www.jeffgeerling.com/blogs/jeff-geerling/controlling-pwr-act-leds-raspberry-pi>'_.

edit:

```
sudo nano /boot/config.txt
```

and add the following at the end:

```
# Disable the ACT LED on the Pi Zero.
dtparam=act_led_trigger=none
dtparam=act_led_activelow=on
```

10.4 Convert USB port to USB1.1

DO NOT DO THIS if you want to use a LAN connection or USB connected storage (usb memory stick or usb conencted disc drive.

It is no problem for the wifi as that does not use the usb interface

edit:

```
sudo nano /boot/cmdline.txt
```

and add the line:

```
dwc_otg.speed=1
```