Crowsnest V4 Guide

YouTube Video #51

TBD: @Juergen2102 @Juergen2102 • vor 5 Minuten @dr.klipper bei einer anderen Kamera kann man sich mit diesem Befehl - v4l2-ctl -d /dev/video0 –list-ctrls - sämtliche Funktionen anzeigen lassen und dann den Fokus manuell in der Crownest.cfg einstellen, mit z.B. v4l2ctl: focus_auto=0,focus_absolute=80

Sponsor

Vielen Dank an Sven van Beckum von 3D Partner (https://partner-3d.de/) für das Sponsoring der Raspberry Pi V3 Cam!

Übersicht

CrowsNest	Kamera Typ	Stream Dienst	Hinweise
Version 3	Raspberry Pi Cam V1 / V2	ustreamer (Legacy)	Einbindung der Kamera normal über den /dev/video**XX** Pfad
	Raspberry Pi Cam V3	*nicht unterstützt*	

CrowsNest	Kamera Typ	Stream Dienst	Hinweise
	ArduCam	*nicht unterstützt*	
	USB Kamera	ustreamer (Legacy)	Einbindung der Kamera normal über den /dev/video**XX** Pfad Voraussetzung ist, dass die Kamera vom System erkannt wird
Version 4	Raspberry Pi Cam V1 / V2	camera-streamer	siehe Raspberry Pi Cam
	Raspberry Pi Cam V3	camera-streamer	siehe Raspberry Pi Cam
	ArduCam	camera-streamer	siehe ArduCam
	USB Kamera	camera-streamer	Einbindung der Kamera normal über den /dev/**XX** Pfad Voraussetzung ist, dass die Kamera vom System erkannt wird siehe USB Cam
Version 4 (andere SBCs, X86 / X64)		ustreamer (Legacy)	

> **Hinweis:**
> Bei älteren Systemen mit Debian Buster ist ein Update auf Bullseye erforderlich!

Was ist neu?

Die größten Neuerungen in Crowsnest V4 sind folgende:

- Crowsnest V4 nutzt einen anderen Streaming Service - camera-streamer. Dieser Service läuft derzeit nur auf Raspberry PIs, soll aber auch andere SBCs erweitert werden.
  camera-streamer ist nochmal deutlich Ressourcen schonender als ustreamer.
- Wird der neue Mode camera-streamer verwendet, dann wird auf einem Raspberry Pi ein neues Kamera Interface (libcamera) verwendet. Erst damit ist es möglich eine Raspberry Pi V3 Cam oder eine ArduCam mit Crowsnest zu betreiben!
- Die Pfade (/dev/video...) zu den Raspberry Pi und ArduCams ändern sich. Hier muss definitiv die Konfiguration angepasst werden (wenn camera-streamer verwendet wird).
- Crowsnest V4 unterstützt WebRTC. Vereinfacht gesagt können damit die Videodaten schneller und effizienter zum Browser übertragen werden. Das führt zu weitaus flüssigeren Videowiedergaben im Web.
- Crowsnest V4 unterstützt RTSP-Streams. Damit lassen sich die Cams z.B. in VLC zusätzlich darstellen.

Update V3 -> V4

> **Hinweis 1**
> Ab MainSail OS 1.2.0 ist Crowsnest V4 vorinstalliert!
>
> Da könnt ihr euch den ganzen Update Kram also sparen 🙂
>
> **Hinweis 2**
> Das Update erfolgt in der SSH Konsole und nicht in MainSail!
> **Crowsnest V4 lässt sich nicht direkt über den UPDATE Button in Mainsail updaten!**

> Wenn ihr das angeklickt habt wird Crowsnest sehr wahrscheinlich nicht sauber laufen.
> Siehe dazu auch den Hinweis bei Crowsnest V3 deinstallieren.

- System updaten
  `sudo apt update && sudo apt upgrade -y`
- **Crowsnest V3 deinstallieren**
  - aktuelle Version im MainSail Update Manager zeigt **Version 3** → ❌
    ```
    cd ~/crowsnest
    make uninstall
    ```
  - aktuelle Version im MainSail Update Manager zeigt **Version 4** weil ihr den **UPDATE Button schon gedrückt** habt
    ```
    cd ~/crowsnest
    git checkout legacy/v3
    make uninstall
    ```
  - Die Fragen bei `make uninstall` wie folgt beantworten:
    ❌
- **Update Crowsnest**
  - `cd ~/crowsnest`
  - `git pull origin master`
  - `git checkout master`
  - `sudo make install`
    - Das dauert jetzt erstmal ne Weile ....
    - Die Frage `Do you want to add 'update manager' entry to your moonraker.conf` mit y beantworten
    - Reboot → y
- **Update moonraker.conf**
  - moonraker.conf in MainSail öffnen
  - Folgende Zeile unterhalb `[update_manager crowsnest]` anpassen:
    `install_script: tools/install.sh` → `install_script: tools/pkglist.sh`
  - Save & Restart

Konfiguration Pi

- **/boot/config.txt anpassen**
  - `sudo nano /boot/config.txt`
  - In der Datei müssen einige Anpassungen gemacht werden:
    - **`camera_auto_detect`** diesen Wert auf 1 setzen bzw. einfügen wenn nicht vorhanden → `camera_auto_detect=1`
    - **`start_x=1`** diese Zeile auskommentieren → `#start_x=1`
    - **`dtoverlay=vc4-kms-v3d`** wenn nicht vorhanden diese Zeile ergänzen. Ohne diese Zeile läuft keine Raspberry Pi Cam. (Siehe dazu auch https://www.raspberrypi.com/documentation/computers/camera_software.html)
  - Den nano Editor verlassen mittels STRG + X → Y → Enter
  - `sudo reboot`

Kamera Test

> **Hinweis**
> **Dieser Test funktioniert nur mit einem laufenden X Server** (ich nutze dafür MobaXTerm)

- `sudo systemctl stop crowsnest.service`
- `sudo apt install -y cheese`
- Den Pfad der Kamera ermitteln
    - Raspberry PI Cams
    - ArduCams
    - USB Kameras
- `cheese /base/soc/i2c0mux/i2c@1/imx708@1a`
- `sudo systemctl start crowsnest.service`
- ☒

Kamera Infos lesen

Pi Cams

Damit die Raspberry Pi Kameras korrekt in der crowsnest.conf eingetragen werden können, brauchen wir den korrekten Pfad zum Gerät.
Dies kann u.a. aus dem crowsnest Log ausgelesen werden. Einfacher geht es aber mit folgendem Befehl in der Konsole:

```
libcamera-vid --list-cameras
```

Als Ergebniss bekommt man dann folgende Ausgaben für die unterschiedlichen Raspberry Pi Cams:

Cam V1 suchen (ov5647 Chip)

```
pi@TestPi3:~ $ libcamera-vid --list-cameras
Available cameras
-----------------
0 : ov5647 [2592x1944] (/base/soc/i2c0mux/i2c@1/ov5647@36)
    Modes: 'SGBRG10_CSI2P' : 640x480 [58.92 fps - (16, 0)/2560x1920 crop]
                             1296x972 [43.25 fps - (0, 0)/2592x1944 crop]
                             1920x1080 [30.62 fps - (348, 434)/1928x1080
crop]
                             2592x1944 [15.63 fps - (0, 0)/2592x1944 crop]
```

Cam V2 suchen (imx219 Chip)

```
pi@PiTest:/dev $ libcamera-vid --list-cameras
Available cameras
-----------------
0 : imx219 [3280x2464] (/base/soc/i2c0mux/i2c@1/imx219@10)
    Modes: 'SRGGB10_CSI2P' : 640x480 [103.33 fps - (1000, 752)/1280x960
crop]
                             1640x1232 [41.85 fps - (0, 0)/3280x2464 crop]
                             1920x1080 [47.57 fps - (680, 692)/1920x1080
crop]
```

```
                              3280x2464 [21.19 fps - (0, 0)/3280x2464 crop]
          'SRGGB8' : 640x480 [103.33 fps - (1000, 752)/1280x960 crop]
                     1640x1232 [41.85 fps - (0, 0)/3280x2464 crop]
                     1920x1080 [47.57 fps - (680, 692)/1920x1080 crop]
                     3280x2464 [21.19 fps - (0, 0)/3280x2464 crop]
```

Cam V3 suchen (imx708 Chip)

```
pi@PiTest:~ $ libcamera-vid --list-cameras
Available cameras
-----------------
0 : imx708_wide_noir [4608x2592] (/base/soc/i2c0mux/i2c@1/imx708@1a)
    Modes: 'SRGGB10_CSI2P' : 1536x864 [120.13 fps - (768, 432)/3072x1728
crop]
                              2304x1296 [56.03 fps - (0, 0)/4608x2592 crop]
                              4608x2592 [14.35 fps - (0, 0)/4608x2592 crop]
```

Von dieser Ausgabe brauchen wir die Angabe die mit `/base` startet. Im Falle der Pi Cam V3 bei mir z.B. `/base/soc/i2c0mux/i2c@1/imx708@1a`.
Hier kann man sich auch gleich die Auflösung und mögliche FPS notieren die wir ebenfalls in der corwsnest.conf später eintragen müssen.

ArduCam

ArduCams sollen mit Crowsnest V4 auch laufen. Mangels Hardware konnte ich das noch nicht testen. Die Einrichtung und Pfad Suche dürfte aber sehr ähnlich wie bei der Raspberry Pi V3 Cam sein.

USB Cam

Wie man den korrekten Device Pfad für USB Kameras ermittelt habe ich schon in einem eigenen Video erklärt:

- http://www.drklipper.de/doku.php?id=videos:41_-_linux_faq_-_webcam_auslesen_und_testen
- https://youtu.be/H1lGdexbXGk

Kurzfassung

- bekannte Video Geräte auflisten
  `v4l2-ctl --list-devices`
- Je nach Kamera bekommt man dann eine Ausgabe wie die folgende:

  ```
  pi@TestPi3:~ $ v4l2-ctl --list-devices

  ...

  USB Camera: USB Camera (usb-3f980000.usb-1.5):
          /dev/video0
  ```

```
        /dev/video1
        /dev/media4
```

- Im Normalfall ist es dann der erste Pfad - hier wäre es also `/dev/video0`
- Die möglichen Formate lassen sich dann so ermitteln:
  `v4l2-ctl --device /dev/video0 --list-formats-ext`
- Und da sich der Pfad `/dev/video0` durchaus ändern kann, bestimmt man besser noch die symbolischen Links zu dem Gerät:
  `udevadm info --root --query=symlink --name=/dev/video0`

```
pi@TestPi3:~ $ udevadm info --root --query=symlink --name=/dev/video0
/dev/v4l/by-path/platform-3f980000.usb-usb-0:1.5:1.0-video-index0
/dev/v4l/by-id/usb-ICT-TEK_USB_Camera_202001010001-video-index0
```

- In diesem Fall würde man also in der crowsnest Konfig eintragen `device: /dev/v4l/by-id/usb-ICT-TEK_USB_Camera_202001010001-video-index0`

crowsnest.log

Neben dem manuellen Auslesen der Kamera Pfade / Infos gibt es auch noch das crowsnest Log. Auch hier kann man die meisten Informationen zu den Kameras finden. Der Vollständigkeit halber hier die beiden Möglichkeiten, die Informationen einzusehen:

- Log komplett einsehen
  `nano ~/printer_data/logs/crowsnest.log`
- Kamera Infos ausgeben
  `~/crowsnest/tools/dev-helper.sh -c`
  Hinweis : dev-helper.sh liefert derzeit nicht den richtigen Gerätepfad für Raspberry Pi Cams! Also da lieber die manuelle Methode nutzen.

crowsnest.conf anpassen

- crowsnest.conf in MainSail öffnen
- `delete_log: false` → `delete_log: true`
  Das macht das Logfile übersichtlicher ...
- In der Demo Config gibt es schon einen Eintrag mit `[cam 1]`. Wenn ihr mehrere Kameras angeschlossen habt, dann diesen ganzen Block erstmal so oft anlegen wie ihr Kameras habt. Der zweite Kamera Block heißt dann `[cam 2]` usw.
- Dann schon mal in allen `[cam ..]` Blöcken den mode umstellen von `ustreamer` auf `camera-streamer`.
- Den Eintrag `port` für jeden cam Eintrag könnt ihr auch anpassen.
  - `[cam 1]` → `port: 8080`
  - `[cam 2]` → `port: 8081`
  - `[cam 3]` → `port: 8082`
  - usw ...
- Den Eintrag `rtsp_port` für jeden cam Eintrag könnt ihr auch anpassen.
  - `[cam 1]` → `port: 8554`
  - `[cam 2]` → `port: 8555`
  - `[cam 3]` → `port: 8556`

    - usw …
- Wenn ihr RTSP verwenden wollt, könnt ihr das über `enable_rtsp: true` aktivieren. RTSP kann man z.B. dafür verwenden, um mit VLC auf die Kamera Streams zuzugreifen.
- `device`, `resolution` und `max_fps` müsst ihr für jede Cam konfigurieren. (Siehe Kamera Infos lesen)
- Jetzt erstmal Save & Restart.

> **Hinweis: Auflösungen und FPS**
> Der Raspberry Pi Video De-/Encoder kann maximal eine Auflösung von 1920×1080 verarbeiten. Auch wenn ihr die Auflösung größer einstellt wird nicht mehr ausgegeben. Zudem verringern sich die möglichen Bilder pro Sekunde (FPS) wenn mehrere Kameras verwendet werden (Erfahrungen Stefan Dej : 2x 1920x1080f30 ⇒ 1920x1080f15 / 2x 1280x720f30 ⇒ 1280x720f25). Dies ist ein Hardware Limit vom Raspberry Pi.

Siehe auch die Beispiel Konfig für Crowsnest V4 am Ende dieses Beitrags.

Web Test

MainSail Konfiguration

Damit die Kameras auch in Mainsail angezeigt werden müssen die hier auch konfiguriert werden.

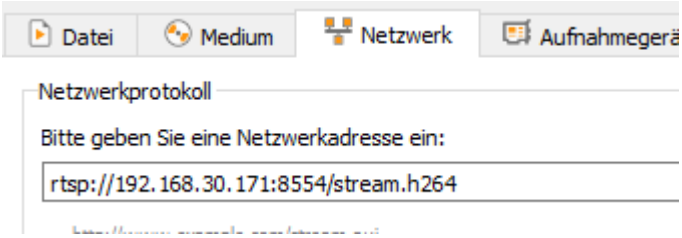
- Oben rechts auf die Zahnräder klicken
- Unter Interface Settings auf WEBCAMS klicken
- Auf ADD WEBCAM klicken
- Kamera Infos angeben
    - Name → Was ihr wollt
    - URL Stream → `/webcam/webrtc`
    - URL Snapshot → `/webcam/?action=snapshot`
    - Service → WebRTC (camera-streamer)
    - UPDATE WEBCAM anklicken
- **HINWEIS** : Mehrere Kameras einstellen
  Wenn ihr mehrere Kameras angeschlossen habt, dann müsst ihr ab der zweiten Kamera die Links anpassen. Aus `/webcam` wird dann für die zweite Cam `/webcam2`, für die dritte `/webcam3` usw.
  Bei mir schaut das dann für die zweite Cam so aus:
- Sollten die Kameras nicht in der Oberfläche angezeigt werden, hilft öfter mal einfach ein Neustart von Crowsnest oder vom Pi selber.

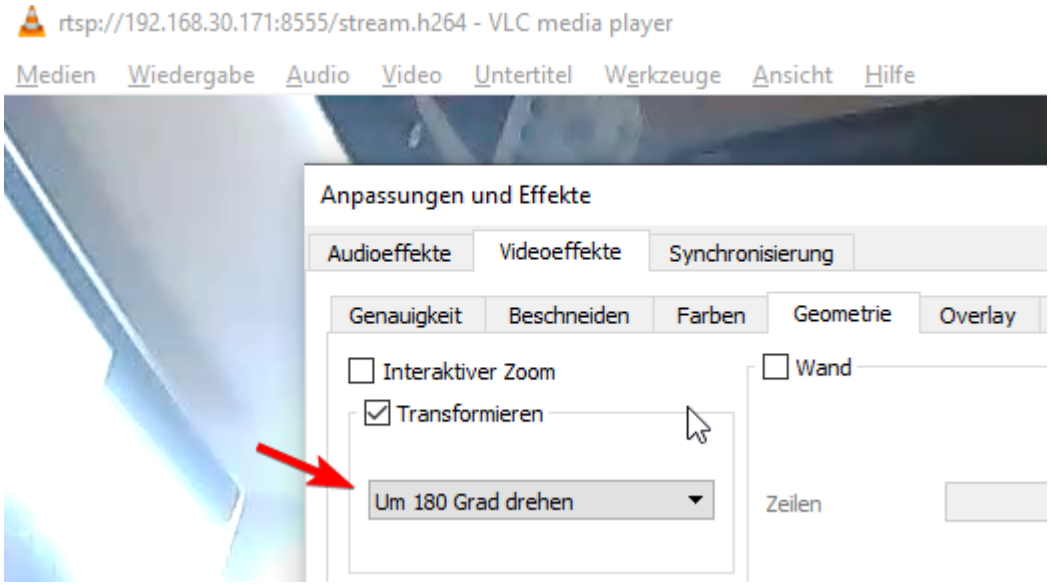
RTSP Stream (VLC)

Die RTSP Streams kann man sich (sofern in der Konfig eingeschaltet) u.a. mit VLC ansehen. Dazu braucht man die IP und den RTSP Port (der ebenfalls in der Konfig eingestellt ist).

Aufrufen kann man das dann wie folgt:

- In VLC auf Medien → Netzwerkstream öffnen ...
- Hier den Pfad eintragen
  - ohne User und Passwort
    `rtsp://<IP>:<PORT>/stream.h264`
  - mit User und Passwort
    `rtsp://<USER>:<PASSWORD>@<IP>:<PORT>/stream.h264`
  - Beispiel



- Sollte das Bild auf dem Kopf stehen, kann man das Bild drehen in VLC
  - Werkzeuge → Effekte und Filter
  - oben auf Videoeffekte
  - darunter auf Geometrie
  - 

Update Crowsnest (Recompile)

- `cd ~/crowsnest/bin/camera-streamer`
- `git pull`
- `cd ~/crowsnest`
- `make buildclean`
- `make build -j4`

Links

- Crowsnest Github
  https://github.com/mainsail-crew/crowsnest
- Crowsnest Doku

https://crowsnest.mainsail.xyz/faq/
- Update V3 → V4
  https://crowsnest.mainsail.xyz/faq/upgrade-from-v3-to-v4
- Raspberry Pi Doku
  https://www.raspberrypi.com/documentation/computers/camera_software.html
- camera streamer
  https://github.com/ayufan/camera-streamer
- Crowsnest is broken after system packages upgrade (Recompile Crowsnest)
  https://github.com/mainsail-crew/crowsnest/issues/143
- Webcams für den Raspberry Pi
  https://elinux.org/RPi_USB_Webcams

Configs

/boot/config.txt

| config.txt

```
## LEGACY Mode    ##
## Disable libcamera (interferes with ustreamer, when using raspicams)
#camera_auto_detect=0
## Enable VideoCore at boot, needed for Crowsnest (Raspicams and DSI
devices).
#start_x=1

## libcamera Mode ##
dtoverlay=vc4-kms-v3d
# CAM V1
#dtoverlay=ov5647
# CAM V2
#dtoverlay=imx219
# CAM V3
#dtoverlay=imx708
# AutoDetect
camera_auto_detect=1
```

crowsnest.conf V3

crowsnest.conf

```
#### crowsnest.conf
#### This is a typical default config.
#### Also used as default in mainsail / MainsailOS
#### See:
#### https://github.com/mainsail-crew/crowsnest/blob/master/README.md
#### for details to configure to your needs.
```

```
#####################################################################
####                                                             #####
####      Information about ports and according URL's            #####
####                                                             #####
#####################################################################
####                                                             #####
####    Port 8080 equals /webcam/?action=[stream/snapshot]       #####
####    Port 8081 equals /webcam2/?action=[stream/snapshot]      #####
####    Port 8082 equals /webcam3/?action=[stream/snapshot]      #####
####    Port 8083 equals /webcam4/?action=[stream/snapshot]      #####
####                                                             #####
#####################################################################


[crowsnest]
log_path: ~/printer_data/logs/crowsnest.log
log_level: verbose                      # Valid Options are
quiet/verbose/debug
delete_log: true                        # Deletes log on every restart,
if set to true

[cam 1]
mode: mjpg                              # mjpg/rtsp
port: 8080                              # Port
device: /dev/video0                     # See Log for available ...
resolution: 640x480                     # widthxheight format
max_fps: 15                             # If Hardware Supports this it
will be forced, otherwise ignored/coerced.
#custom_flags:                          # You can run the Stream
Services with custom flags.
#v4l2ctl:                               # Add v4l2-ctl parameters to
setup your camera, see Log what your cam is capable of.

[cam 2]
mode: mjpg                              # mjpg/rtsp
port: 8081                              # Port
device: /dev/video1                     # See Log for available ...
resolution: 640x480                     # widthxheight format
max_fps: 15                             # If Hardware Supports this it
will be forced, otherwise ignored/coerced.
#custom_flags:                          # You can run the Stream
Services with custom flags.
#v4l2ctl:                               # Add v4l2-ctl parameters to
setup your camera, see Log what your cam is capable of.
```

crowsnest.conf V4

crowsnest.conf

```
#### crowsnest.conf
#### This is a typical default config.
#### Also used as default in mainsail / MainsailOS
#### See:
#### https://github.com/mainsail-crew/crowsnest/blob/master/README.md
#### for details to configure to your needs.


######################################################################
####                                                              #####
####      Information about ports and according URL's             #####
####                                                              #####
######################################################################
####                                                              #####
####    Port 8080 equals /webcam/?action=[stream/snapshot]        #####
####    Port 8081 equals /webcam2/?action=[stream/snapshot]       #####
####    Port 8082 equals /webcam3/?action=[stream/snapshot]       #####
####    Port 8083 equals /webcam4/?action=[stream/snapshot]       #####
####                                                              #####
######################################################################
####    RTSP Stream URL: ( if enabled and supported )             #####
####    rtsp://<ip>:<rtsp_port>/stream.h264                       #####
######################################################################


[crowsnest]
log_path: /home/pi/printer_data/logs/crowsnest.log
log_level: verbose                      # Valid Options are
quiet/verbose/debug
delete_log: true                       # Deletes log on every restart,
if set to true
no_proxy: false

[cam 1]
mode: camera-streamer                         # ustreamer - Provides
mjpg and snapshots. (All devices)
                                        # camera-streamer - Provides
webrtc, mjpg and snapshots. (rpi + Raspi OS based only)
enable_rtsp: true                      # If camera-streamer is used,
this enables also usage of an rtsp server
rtsp_port: 8554                         # Set different ports for each
device!
port: 8080                              # Port
#device: /base/soc/i2c0mux/i2c@1/imx708@1a                    # See
Log for available ...
device: /base/soc/i2c0mux/i2c@1/ov5647@36
resolution: 640x480                     # widthxheight format
max_fps: 25                             # If Hardware Supports this it
will be forced, otherwise ignored/coerced.
#custom_flags:                          # You can run the Stream
Services with custom flags.
```

```
#v4l2ctl:                                # Add v4l2-ctl parameters to
setup your camera, see Log what your cam is capable of.

[cam 2]
mode: camera-streamer                         # ustreamer - Provides
mjpg and snapshots. (All devices)
                                         # camera-streamer - Provides
webrtc, mjpg and snapshots. (rpi + Raspi OS based only)
enable_rtsp: true                       # If camera-streamer is used,
this enables also usage of an rtsp server
rtsp_port: 8555                          # Set different ports for each
device!
port: 8081                               # Port
device: /dev/v4l/by-id/usb-SIT_USB2.0_Camera_SIT_USB2.0_Camera-video-
index0                     # See Log for available ...
resolution: 640x480                      # widthxheight format
max_fps: 25                              # If Hardware Supports this it
will be forced, otherwise ignored/coerced.
#custom_flags:                           # You can run the Stream
Services with custom flags.
#v4l2ctl:                                # Add v4l2-ctl parameters to
setup your camera, see Log what your cam is capable of.
```

From:
https://drklipper.de/ - **Dr. Klipper Wiki**

Permanent link:
https://drklipper.de/doku.php?id=klipper_faq:51_-_crowsnest_v4_guide

Last update: **2024/06/29 06:17**