Rover Cam Manual

Rover Cam is a network independent security, and pest control software, to be used on Raspberry Pi.

On motion detection it can:

- record clips, or images
- play one or more audio files, or run python files
- control up to 8 GPIO pins by turning them on and off, called “Actions”

With extra electronics and hardware, Rover Cam can be used for pest control, by making sounds that frighten, or irritate animals, by moving an object to scare the animal, or initiating a spray of water, as motion sensing sprinklers do. As a security camera it can sound an alarm, or any custom audio recording, and/or move objects.

System Requirements (version tested on):

Raspberry Pi OS (32 bit)

- Python (3.7.3)
- opencv-contrib-python (4.5.3)
- numpy (1.21.2)
- Pillow (5.4.1)
- RPi.GPIO
- gpiozero
- Raspberry Pi 4

Recommended:

- 3 buttons with connectors and 1k ohm resistors
- At least 1 5mm LED with 1k ohm resistor to serve as indicator light
- A USB storage device

Installation

If Rover Cam is downloaded with OS, no further installation is required. Can be moved to USB storage, but crontab will need to be edited, if using “run at system boot” option.

If Rover Cam is not downloaded with the OS

1. Edit crontab file for launching “start-up.py” file at boot. Edit as user, if done as root audio won’t work.

To access crontab:

```
crontab -e
```

Open with nano if prompted.

At bottom put:

@reboot sleep 40 && python3 #path to the Run-at-startup folder#/*.py &

Example:
@reboot sleep 40 && python3 /home/pi/Desktop/Rover_Cam/Run-at-startup/*.py &

Example if running from USB:
@reboot sleep 40 && python3 /media/pi/KINGSTON/Rover_Cam/Run-at-startup/*.py &

"&" is important, it's also important that file does not fault out, pi may not boot. The file is ran in a "Try" block so that might limit this risk, but it hasn't been tested.

2. In "/boot/config.txt", uncomment the "#hdmi_force_hotplug=1" by removing the #. This allows the pi to run properly without a monitor.

GPIO Set-Up

Button Hook-Up

Using the GPIO BCM numbering, and a 1k ohm resistor for each hook-up. Shared or any ground.

Power Button (red)	Pin 17
Mode Select Button (gray)	Pin 27
Disable Button (blue)	Pin 22

Action Hook-Up

Action 1	Pin 12
Action 2	Pin 16
Action 3	Pin 20
Action 4	Pin 21
Action 5	Pin 23
Action 6	Pin 24
Action 7	Pin 13
Action 8	Pin 19

Indicator LED	Pin 26

Operation

The Settings Interface interacts with RC_config.txt, and aoi_pts.txt. These files can be edited on any type of operating system.

To Initiate Settings Interface:
In terminal:

python3 #path to#/Rover_Cam.py
Or launch it with an IDE

To Initiate Normal Operation:
In Settings turn on “Run At System Boot”
Restart the pi
Or run start-up.py from terminal, or an IDE

Setting Date Time

Rover Cam operates independently of any networks, so if accurate date time stamps are desired on video clips, date time will have to be set for each boot up, in the settings interface, or config.txt. Set the date time to about 1 minute after the pi will be turned on with “Run At System Boot” on. This will give the pi time to boot up, and the software to initiate a timer.

States

When power is supplied to the pi and “Run At System Boot” is on, after about a minute the camera, will be active and any Actions, audio, and clip recording will be initiated on motion detection. From here on there are 3 possible states that Rover Cam can be in:

Operational State – this is where if goes as described above, one of 3 modes are running as chosen in settings.

Mode Selection State – the camera and motion detection are not operational the available modes can be toggled through, as an easy “in the field” way to switch modes.

Disabled State – this is when the disable code is inputted, and the system is disabled.

Button Operation

When in Operational State:

Button	Goes To / Initiates
Disable Button (blue) Pressed (if code passed)	Disabled State
Mode Select Button (gray) Pressed	Indicator LED blinks slow the number of times corresponding to the mode number. This tells the user that a mode is running and which one. Then blinks fast corresponding to the number of saved clips or images since last start, capped at 30 blinks.
Mode Select Button (gray) Held 3s	Leaves Operational State and enters Mode Selection State, where modes can be selected.
Power Button (red) Pressed	Reboots the pi (in about 20s)
Power Button (red) Held 3s	Shuts down the pi

When in Mode Selection State:

Button	Goes To / Initiates
Disable Button (blue) Pressed (if code passed)	Nothing
Mode Select Button (gray) Pressed	Toggle through available modes. Indicator LED blinks slow the number of times corresponding to the mode number.
Mode Select Button (gray) Held 3s	Leaves Mode Selection State and goes back to where it previously was, either Operational State, or Disabled State.
Power Button (red) Pressed	Reboots the pi (in about 20s)
Power Button (red) Held 3s	Shuts down the pi

When in Disabled State:

Button	Goes To / Initiates
Disable Button (blue) Pressed (if code passed)	Operational State
Mode Select Button (gray) Pressed	Nothing
Mode Select Button (gray) Held 3s	Mode Selection State
Power Button (red) Pressed	Reboots the pi (in about 20s)
Power Button (red) Held 3s	Shuts down the pi

Modes

When in Operational State there are 3 possible modes to choose from:

Mode1 – only records clips, or images on motion
Mode2 – records clip or images, and performs Actions, and plays audio or python files on motion
Mode 3 – only perform Actions/audio or python files on motion

Audio / Python Files

If “Play Audio” is set to “on” and there are audio/python files in “audio” folder then each file in that directory will run one after the other, in alphabetical order. They will not start again until they are finished running, even if “Clip Time” is shorter. If motion is detected another clip or image will be record even if the audio/python files are still running.

See the setting interface for more details.

Actions

Actions should always start in “off” even if for 0 seconds, and not have 2 consecutive “ons” or “offs”. They can end in either “on” or “off”. All Action will begin as the same time, and will not start again until they are all finished. If “Clip Time” is shorter than the longest Action run time, and motion is detected again before all the Actions are finished, Actions will not start on this motion

instance. Setting one Action to end in “off” for an extended period of time is a good way to restrict Actions from starting again while recording images or short clip times.

See the setting interface for more details.

Extra Tip:

In terminal to see processes running:

```
htop
```

htop may show multiply files running but is only one file running on multiple threads (this can be changed in htop set-up/file)

DISCLAIMER:
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.