

ADAMS

Advanced **D**ata mining **A**nd **M**achine learning **S**ystem

Module: adams-video



Peter Reutemann
Steven Brown

January 10, 2024

©2015-2018



THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato



Except where otherwise noted, this work is licensed under
<http://creativecommons.org/licenses/by-sa/4.0/>

Contents

1	Flow	7
2	Tools	9
2.1	Trail viewer	9
2.2	VLCj Video Player	9
2.2.1	Bitness	10
2.2.1.1	Windows	10
2.2.1.2	Linux	10
2.2.1.3	Mac OSX	11
2.3	Annotator	13
2.3.1	Video Menu	13
2.3.2	Annotations Menu	13
2.3.3	Background Menu	13
2.3.4	Extract Background	14
2.3.5	Shortcuts Menu	14
2.3.5.1	Edit Shortcuts	14
3	Data	19
3.1	.trail format	19
	Bibliography	21

List of Figures

2.1	Trail viewer displaying a mice trail overlayed on a background image.	9
2.2	Video Player in use.	10
2.3	Windows 64-bit.	11
2.4	Java 64-bit.	12
2.5	VLC 64-bit.	12
2.6	Annotator showing a video and key shortcuts	13
2.7	Video menu.	14
2.8	Video menu.	15
2.9	Annotations menu.	16
2.10	Background menu.	16
2.11	Extract Background Dialog	17
2.12	Image Sampler Selection	17
2.13	Shortcuts menu.	18
2.14	Edit shortcuts dialog.	18
2.15	Dialog for editing a shortcut.	18

Chapter 1

Flow

The video module offers some actors for basic video display and processing support.

Available standalones:

- *RecordingSetup* – configures a recording setup (sound/screen/webcam).
- *StartRecording* – starts a defined recording setup.
- *StopRecording* – stops a defined recording setup.

Available sources:

- *ListWebcams* – lists the names of all available webcams attached to the computer¹.
- *WebcamImage* – outputs images from the selected webcam attached to the computer².
- *WebcamInfo* – outputs images from the selected webcam ³.

Available transformers:

- *AddTrailBackground* – adds a image as trail background⁴
- *AddTrailStep* – adds an additional step to the trail passing through⁵.
- *ExtractTrackedObject* – extracts a tracked object in an image and forwards it as new image. container⁶.
- *GetTrailBackground* – retrieves the trail background image, if any.
- *MjpegImageSequence* – generates an image sequence from MJPEG movies, one frame at a time⁷.
- *MovieImageSampler* – extracts images from a movie using a sampling algorithm.
- *MovieImageSequence* – generates an image sequence from movies, one frame at a time (uses Xuggle[3]⁸).

¹adams-video-list_webcams.flow

²adams-video-webcam.flow

³adams-video-webcam_info.flow

⁴adams-video-track_objects-predefined3.flow

⁵adams-video-track_objects-predefined3.flow

⁶adams-video-track_objects-user_selected_object.flow

⁷adams-video-play_mjpeg_video.flow

⁸adams-video-play_mp4_video.flow

- *MovieInfo* – extracts information from movie files⁹.
- *TrackObjects* – tracks objects in images sequences, e.g., from movies¹⁰.
- *TrailFileReader* – reads a trail from disk.
- *TrailFileWriter* – writes a trail to disk¹¹.
- *TrailFilter* – applies a filter to the trail passing through.
- *TransformTrackedObject* – transforms a tracked object in an image with a callable transformer, e.g., for blurring a face¹².

Available sinks:

- *AnimatedGifFileWriter* – generates an animated GIF from an array of image files or images.
- *TrailDisplay* – displays trail objects¹³.

Available conversions:

- *QuadrilateralLocationCenter* – outputs a Point2D object that is the center of the rectangle surrounding the quadrilateral coordinates.
- *QuadrilateralLocationToString* – turns the quadrilateral coordinates into a string.
- *StringToQuadrilateralLocation* – turns a string into quadrilateral locations.

⁹adams-video-movie_info.flow

¹⁰adams-video-track_objects-predefined.flow, adams-video-track_objects-predefined2.flow

¹¹adams-video-track_objects-predefined3.flow

¹²adams-video-track_objects-predefined2.flow

¹³adams-video-display_trail.flow, adams-video-track_objects-predefined3.flow

Chapter 2

Tools

2.1 Trail viewer

The *Trail viewer* is a simple tool for viewing trails of objects that have been tracked using a flow. It can be used to apply filters to trails and save those trails back to disk. Figure 2.1 shows a screenshot.

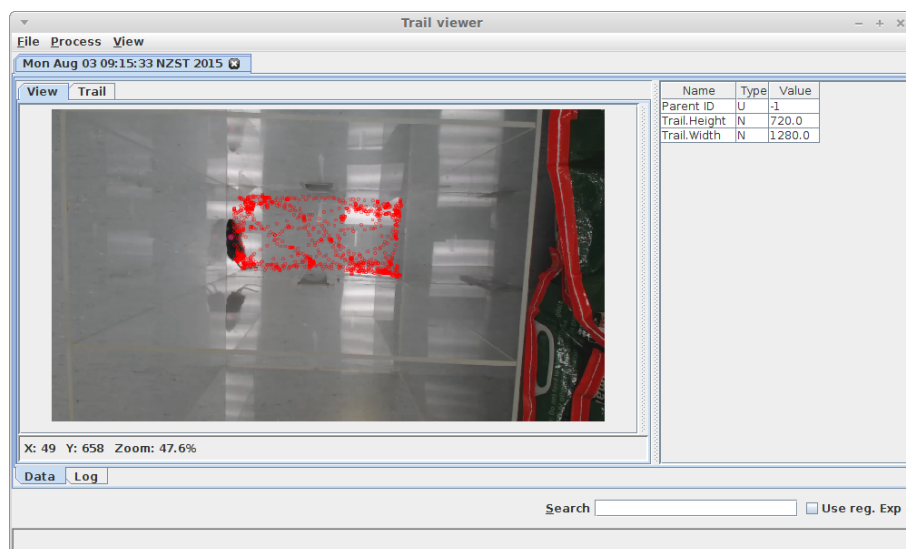


Figure 2.1: Trail viewer displaying a mice trail overlayed on a background image.

2.2 VLCj Video Player

A simple video player used to play video or audio files in real time. To use this tool, or any other tools that incorporate it, a user must have VLC player installed on their computer (see section 2.2.1 for details on correct installation). Figure 2.2 shows the player being used.



Figure 2.2: Video Player in use.

2.2.1 Bitness

The bitness of the VLC player installation and the Java installation must be the same, i.e., they must both either be 32-bit or 64-bit.

2.2.1.1 Windows

In order to determine whether you have a 64-bit version of Windows, you can simply right-click on your computer (could be called *My Computer* or *This PC*) and select *Properties*. If the dialog mentions *64-bit* as in Figure 2.3, then you are indeed on a 64-bit operating system.

To determine the bitness of Java, simply open a command prompt and run the command `java -version`. If the output shows *64-bit* as in Figure 2.4, then you have a 64-bit version installed.

Finally, if you have a 64-bit version of Windows then you will find a *Program Files* and *Program Files (x86)* directory. If you installed a 64-bit version of VLC, it will get installed in the *Program Files* directory (see Figure 2.5), a 32-bit version will end up in the *Program Files (x86)* directory.

Unfortunately, VLC offers the 32-bit version for download by default and you have to explicitly select the 64-bit version.

2.2.1.2 Linux

You can easily determine the bitness of your operating system by executing the following command in a terminal:

```
arch
```

If it states *x86_64*, you have a 64-bit operating system.

Use the following command in a terminal to determine the bitness of your Java installation:

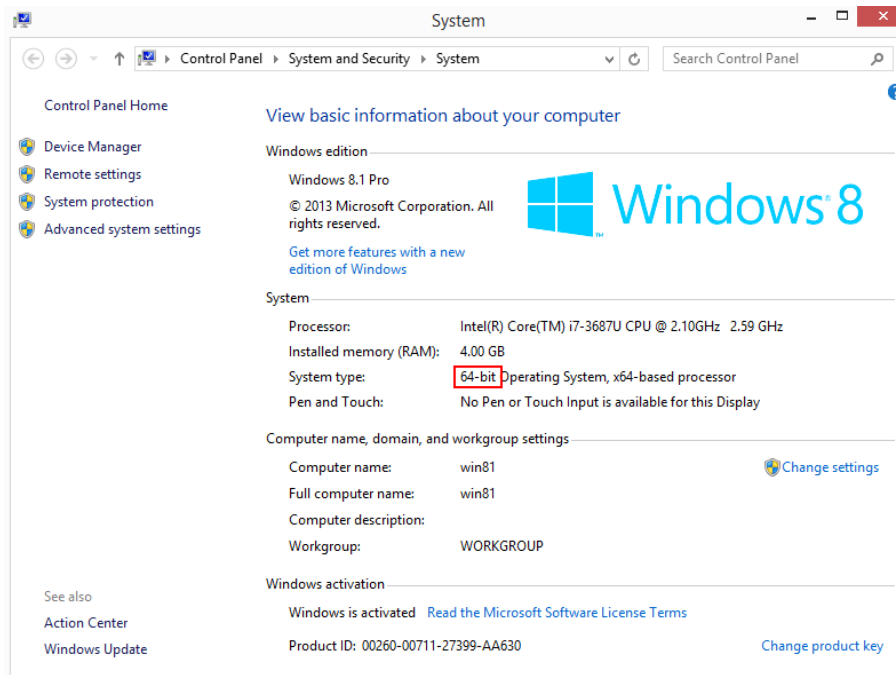


Figure 2.3: Windows 64-bit.

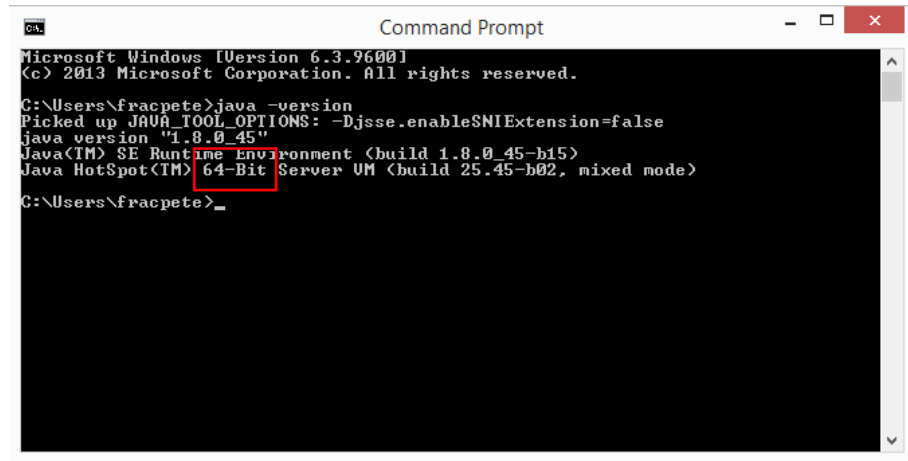
```
java -version
```

It should produce similar output as in Figure 2.4. If it states 64-bit then you have a 64-bit version installed.

Installing VLC through your package manager should already install the same bitness as your operating system.

2.2.1.3 Mac OSX

Mac OSX only comes in 64-bit, so there shouldn't be any issues with the bitness.



```

Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\fracpete>java -version
Picked up JAVA_TOOL_OPTIONS: -Djsse.enableSNIExtension=false
java version "1.8.0_45"
Java(TM) SE Runtime Environment (build 1.8.0_45-b15)
Java HotSpot(TM) 64-Bit Server VM (build 25.45-b02, mixed mode)

C:\Users\fracpete>_

```

Figure 2.4: Java 64-bit.

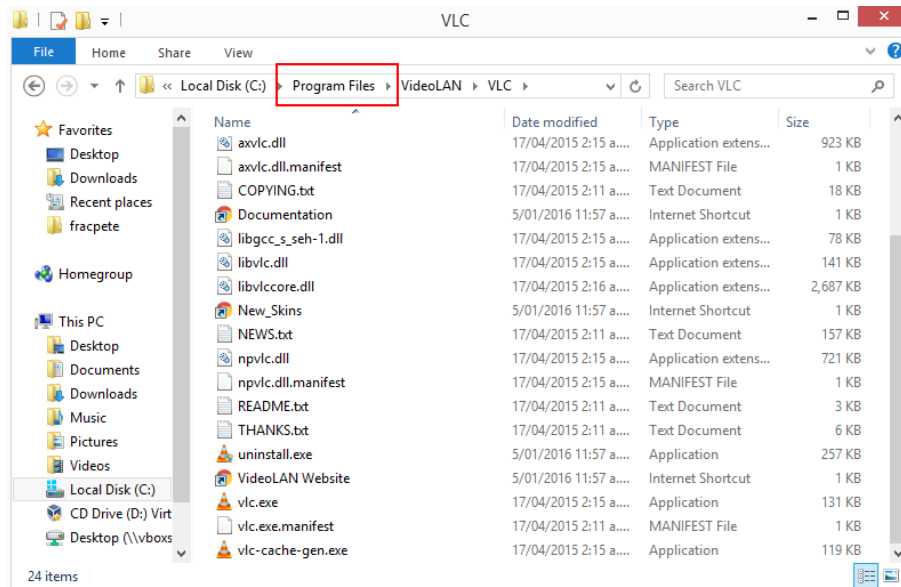


Figure 2.5: VLC 64-bit.

2.3 Annotator

The Annotator tool allows you to annotate videos with meta-data. Just like the *VLCj Video Player*, it uses VLC as well for playing the videos. Therefore please ensure that you setup if correct – see section 2.2.1. Figure 2.6 shows a screenshot of the Annotator in action. The tool has three menus which we will

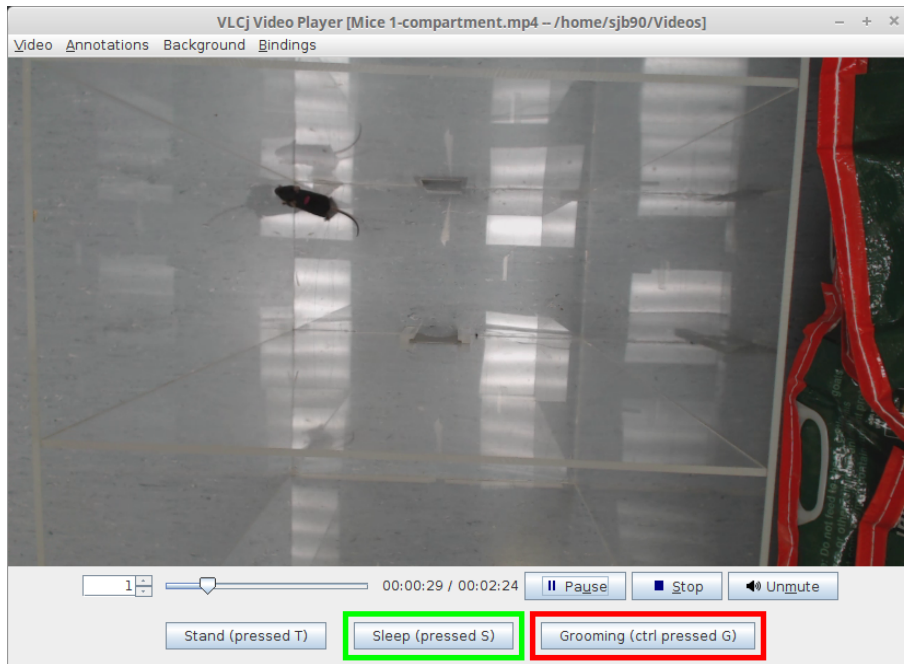


Figure 2.6: Annotator showing a video and key shortcuts

cover in the following sections.

2.3.1 Video Menu

The video menu (see Figure 2.8) lets you open video files or quit the program. It also provides a list of recently opened videos.

2.3.2 Annotations Menu

This menu lets you create a new set of annotations or export the current set to a spreadsheet file, e.g., CSV or, depending on modules present, MS Excel (see Figure 2.9).

2.3.3 Background Menu

The background menu, as depicted in Figure 2.10 allows for the extraction of the background of a video. This requires that the camera shooting the video was fixed. The menu allows you to clear the current background, extract the background from the currently loaded video, save or load the background, and view the current background image.

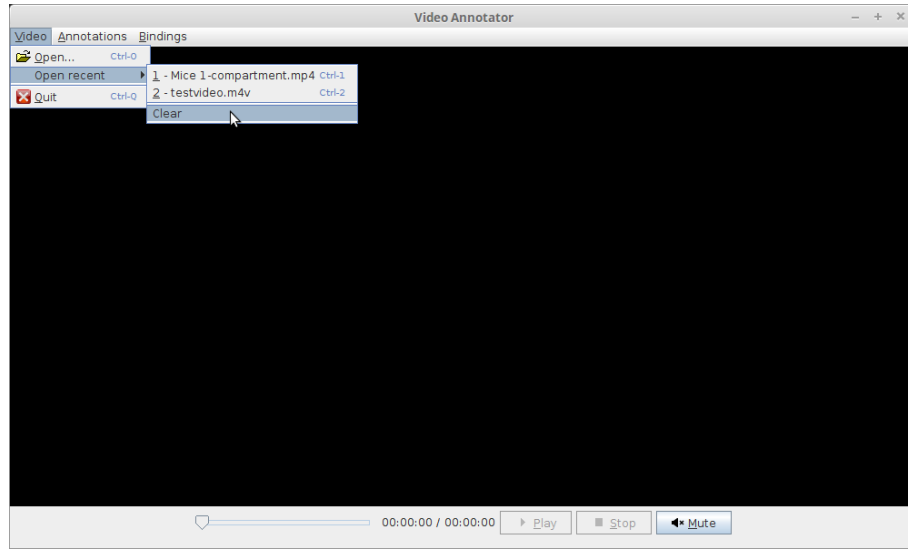


Figure 2.7: Video menu.

2.3.4 Extract Background

Extracting the background is done in the dialog seen in Figure 2.11.

Here a user can select an image sampler and image operation to use to generate the background. Usually the defaults should be sufficient. The selection process can be seen in Figure 2.12.

2.3.5 Shortcuts Menu

The shortcuts menu, as depicted in Figure 2.13, allows for the creation of a new set of shortcuts, loading or saving shortcuts, and editing the current shortcuts.

2.3.5.1 Edit Shortcuts

When editing shortcuts you can add new shortcuts, edit a current shortcut or remove selected shortcuts (see Figure 2.14).

When editing a shortcut there are various options for customizing the output from the shortcut.

The output from a shortcut is made up of a header and *true* or *false* along with a timestamp:

```
Timestamp,Meta-Shortcut1,Meta-Shortcut2
00:00:00.000,false,false
00:00:01.000,false,true
```

In this case the name of the shortcut shows up as *Shortcut1* and *Shortcut2*. Figure 2.15 shows the dialog for editing a shortcut.

In the following, an description of what each of the shortcut parameters is used for:

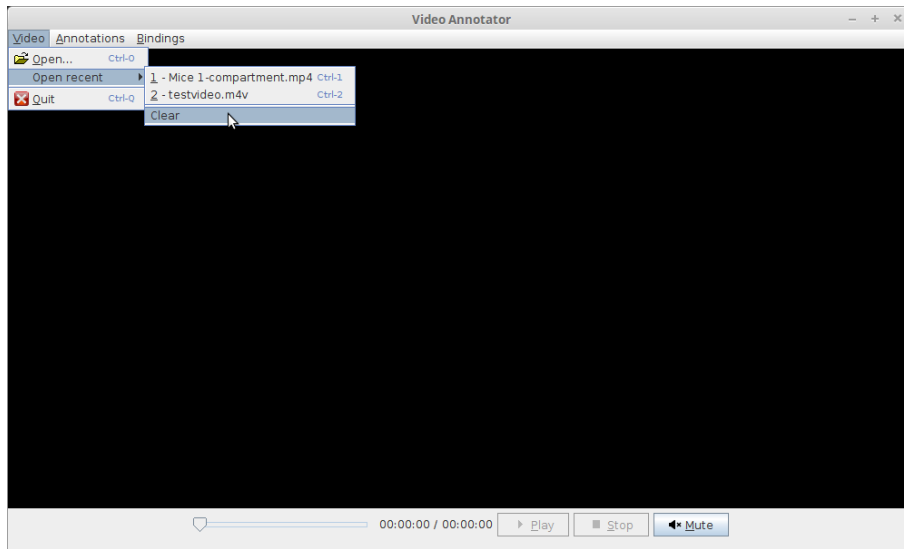


Figure 2.8: Video menu.

- **Name** – This is what gets output as the header of the annotation file when it is exported
- **Shortcut** – The key for this shortcut. It is entered by selecting the text box and then pressing the desired key or key combination – i.e. `ctrl + B` or just `B`
- **Toggleable** – If this is checked then the shortcut always outputs a value, usually *false*, and when it is toggled switches that output to *true*.
- **Interval** – Only relevant for toggleable shortcuts. This is the interval at which the values will be output. By default it is 1,000 milliseconds (= 1 second) but it can be changed to any number. Intervals below 1 second tend to become less accurate, so if set at 10 ms it might output a value between 1 - 20 ms rather than 10 depending on CPU load and scheduling. It is recommended to use times of a second or above.
- **Inverted** – This allows you to invert the values output by the shortcut. Usually a shortcut outputs *true* when pressed or, in the case of toggleable shortcuts, toggled on. This will invert that so when it is pressed it will output *false* and in the case of a toggleable it will output a value of *true* when it is toggled off and *false* when toggled on.

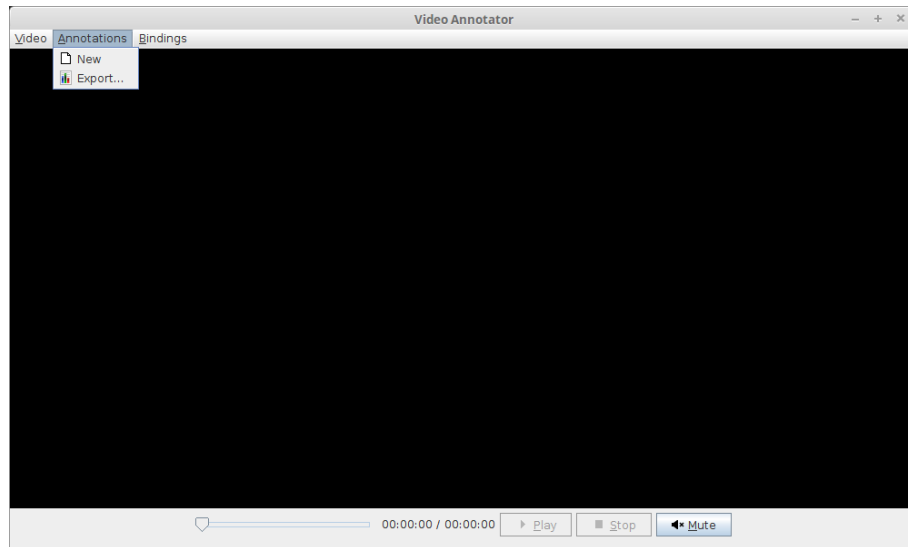


Figure 2.9: Annotations menu.

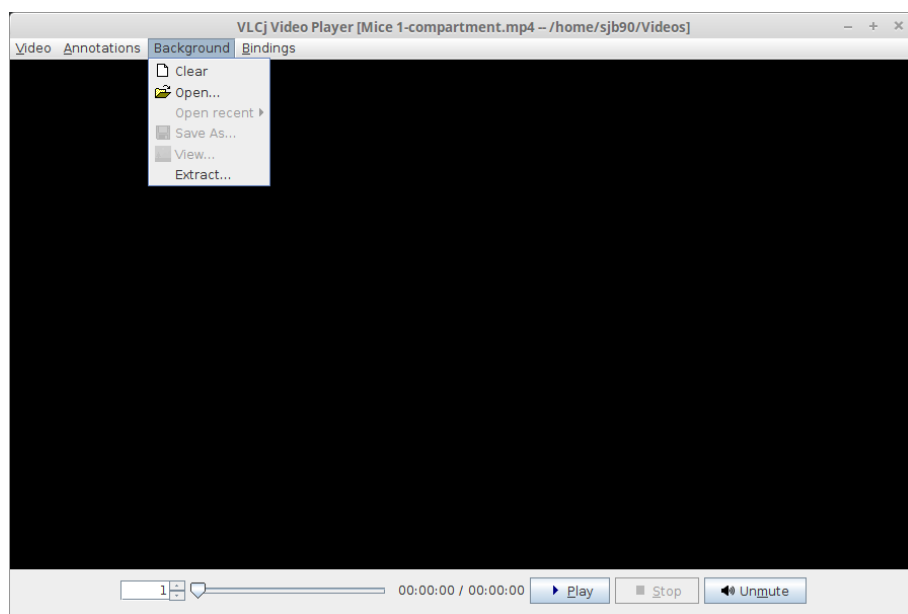


Figure 2.10: Background menu.

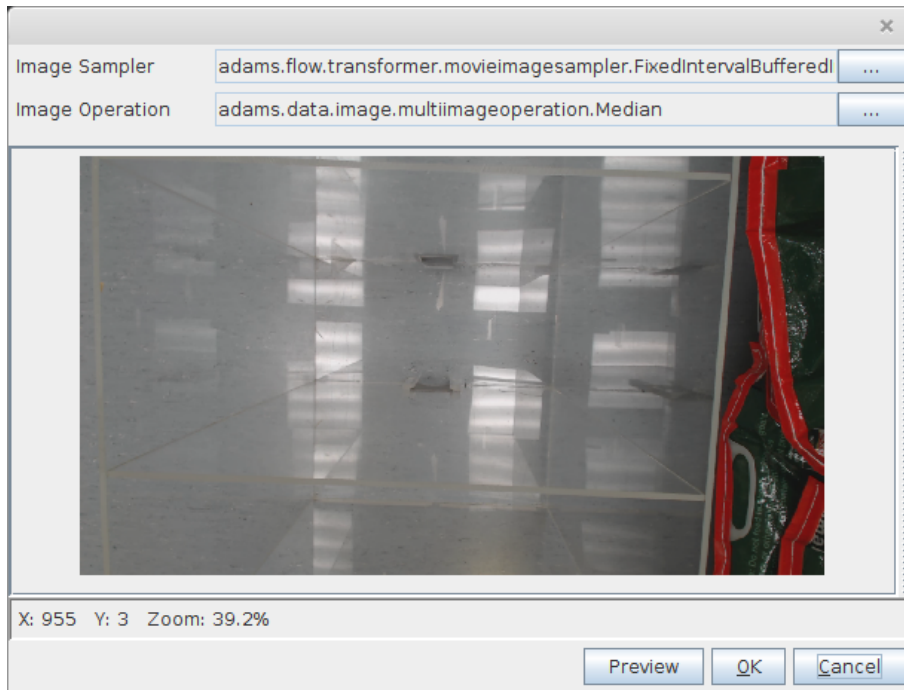


Figure 2.11: Extract Background Dialog

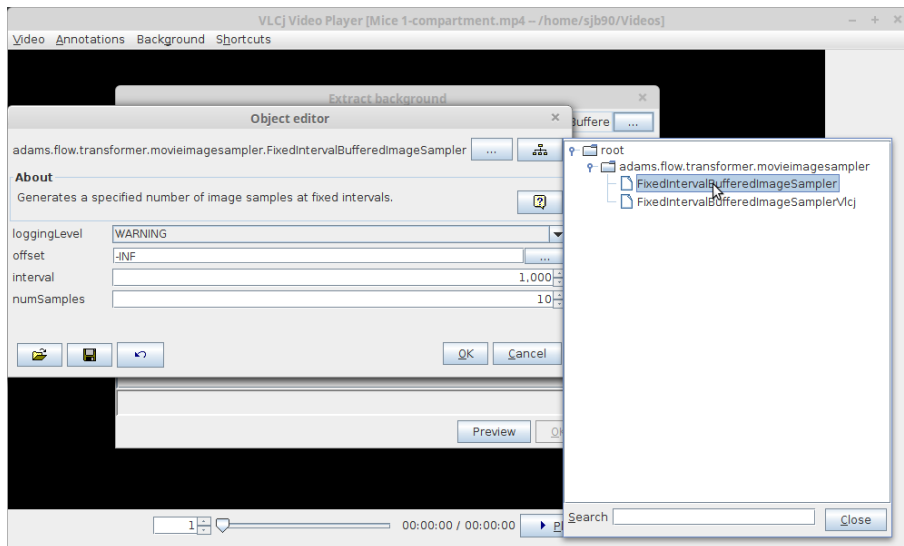


Figure 2.12: Image Sampler Selection

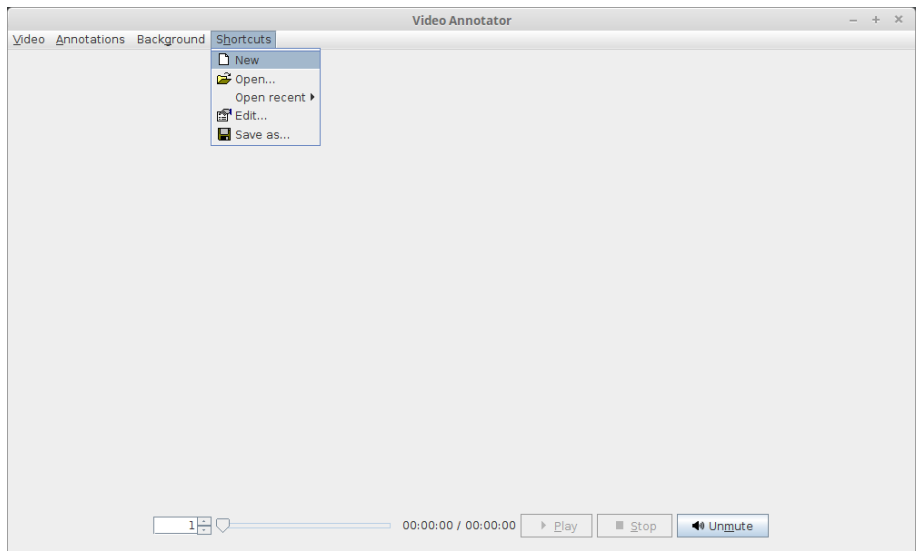


Figure 2.13: Shortcuts menu.

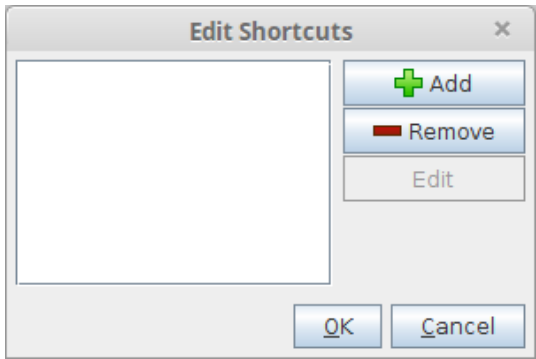


Figure 2.14: Edit shortcuts dialog.

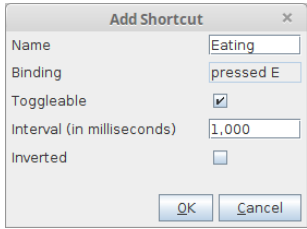


Figure 2.15: Dialog for editing a shortcut.

Chapter 3

Data

3.1 .trail format

The *.trail* format is a simple text format. In its essence it is spreadsheet-like format with a header and a body.

The *header* contains the meta-data for the trail. Most importantly, the width and the height of the trail. The format is in Java Properties format, each line prefixed with “# ”. Each stored value also defines in a separate property what data type it is (N = numeric, S = string, B = boolean).

It is possible to store an optional background image in the header. The image data is stored as RGBA signed bytes, row-by-row. In order to produce smaller files, the data is compressed using `gzip`[4]. The compressed bytes are then stored in a upper-case hexadecimal notation, with a maximum of 1000 bytes per line. The background data lines are prefixed with “% ”.

The *body* consists of four columns: timestamp (with milli-seconds), X position, Y position, meta-data for that step. The meta-data column is a blank-separated list of key-value pairs (“key=value”).

Here is an example file, without a background:

```
# #Tue Jul 28 09:24:20 NZST 2015
# Trail.Height\tDataType=N
# Trail.Width\tDataType=N
# Trail.Height=720.0
# Parent\ ID=-1
# Trail.Width=1280.0
Timestamp,X,Y,Meta-data
"00:00:00.127",424,292,""
"00:00:00.224",423,285,""
"00:00:00.304",423,277,""
```


Bibliography

- [1] *ADAMS* – Advanced Data mining and Machine learning System
<https://adams.cms.waikato.ac.nz/>
- [2] *FFmpeg* – a complete, cross-platform solution to record, convert and stream audio and video
<http://ffmpeg.org/>
- [3] *xuggle* – a free open-source library for Java developers to uncompress, manipulate, and compress recorded or live video in real time
<http://www.xuggle.com/>
- [4] *gzip* – compression/decompression algorithm based on the DEFLATE algorithm, which is a combination of LZ77 and Huffman coding.
<https://en.wikipedia.org/wiki/Gzip>