Discover the Experience

Video Compression Toolbox 2.5

ADTF v2.10.0 or higher required

EB Assist ADTF

Video Compression Toolbox – EB Assist ADTF

Elektrobit Automotive GmbH
Am Wolfsmantel 46
91058 DE-Erlangen, Germany
+49-9131-7701-0
+49-9131-7701-6333
info.automotive@elektrobit.com

Technical support

EB Assist ADTF Support

Phone: +49-9131-7701-7777

http://automotive.elektrobit.com/support

Contents

1 Introduction

1.1 Overview

The *Video Compression Toolbox* adds the ability to handle compressed video streams to ADTF. The following functionalities are added:

- Compressing standard ADTF video samples into compressed *MediaSamples*
- Decompressing compressed *MediaSamples* into standard ADTF video samples.

1.2 Installation

1.2.1 Requirements

The system requirements are described in chapter 1.1 of the ADTFUserManual.

1.2.2 Installation type

It is strongly recommended to install the *Video Compression Toolbox* to `<ADTF_DIR>/addons`. If you install to a different location you have to manually add the `bin` (release) or `bin/debug` (debug) directory to your plugin path (**Options** → **Settings** → **Plugins**).

In versions of the *Video Compression Toolbox* that include services, these must be specified in the main ADTF manifest file when installing the toolbox into a custom location. The main manifest file is located in `<ADTFInstallDir>/bin/)` where any additional service is added below the tag `<manifests>` in the following manner:

```
<manifest optional="false" url="<installdir>/bin/*.manifest" />
```

If you are using ADTF on a Windows 7 operating system, you should not install it into the "ProgramFiles" or any other "System" directory. Because of the user account control (UAC) you have to have administrative privileges to run CMake and/or the batch-file to build the examples. If you don't have these privileges, the build will fail every time.

It is not possible to use the silent installer for automated platform-mixing installations (*Video Compression Toolbox* 32 bit on 64 bit operating systems).

1.2.3 Licensing

The *Video Compression Toolbox* does only run with a valid toolbox flag in your ADTF license. If the needed feature flag is not set, the *Plugins* will not be loaded.

There is no warning that the needed flag could not be found in the given license file while starting ADTF except in the *ADTF DebugMonitor*.

1.3 Additional Hints

Please consider following additional hints:

If you include the supplied `video_compression.manifest` file into the *Manifest File*, do NOT add the *Services* manually to the *Manifest File* again!

If you don't find an installed codec in the *Property Browser*, you can do one of these steps:

- Write the codec name into the *Property* ***Codec*** (e.g. *vfw:XVID*). ADTF will raise a warning in the *ADTF DebugMonitor*.
- Close ADTF, delete the filter cache, and restart *ADTF*. The codec should be shown now. The filter cache can be found in the following directories:
  - for Windows: `C:\Users\<username>\AppData\Roaming\ADTF\fdcache\<adtf_version>`
  - for Linux: `/home/build/.adtf/fdcache/<adtf_version>`

1.3.1 Linux FFmpeg library

On Linux the FFmpeg library is used to support video codecs. Please make sure that you have installed the necessary FFmpeg libraries. Installation instructions for these libraries can be found in the file `Readme.txt` located in the `doc` subdirectory of your ADTF installation.

The *ADTF FFmpeg Support Service* is linked against the FFmpeg library. If FFmpeg is not installaed on your machine then this *Service* will not be loaded and its functionality is not available.

For further license considerations refer to http://www.ffmpeg.org/legal.html.

1.4 New MediaType adtf.type.video_compressed

The *ADTF Video Compression Toolbox* introduces a new ADTF *MediaType* with the class identifier `adtf.type.video_compressed`. the *MajorType* is `MEDIA_TYPE_VIDEO` and the *SubType* is `MEDIA_SUBTYPE_VIDEO_COMPRESSED`.

Both are defined inside the header file `adtf_default_mediatypes.h`.

1.5 Special text formats and symbols

This software guide uses special text formats and symbols to indicate important elements and facts, as shown here:

Windows, **Dialogs** and other **elements of the user interface**

`File names`, `directory names`, etc.

Cross reference section 1.5 Special text formats and symbols

URLs: www.url.com

`inline code`

```
Source code
```

Properties

Proper names

Warnings indicate potential error sources.
→ The arrow indicates the steps you have to take to prevent an error.

This note symbol indicates useful information.

Tips provide additional information.

2 Codec Support

2.1 All Platforms

The ADTF and JPEG codec are supported on all platforms.

Name	Type	Comment
ADTF	lossless	Internal codec of the *ADTF Video Compression Toolbox* which is capable of processing 8/10/12/14/16 bit grayscale images and 24/32 bit color images. Technical details about the implementation: In the first step pixel values are transformed to the difference between their original value and their neighbourhood pixel values. In the second step an entropy coder (*RICE* algorithm) is used to transform the values to bit sequences.
JPEG	lossy	This codec uses the libjpeg-turbo library to compress each frame into a JPEG image. It supports two modes (via the "abbreviated" *Property*): ► Complete: each compressed images conatins all necessary quantization and Huffman tables. ► Abbreviated: the quantization and Huffman tables are stored in the *MediaType* and each compressed image contains image data only. This mode is a little bit faster and more efficient at the cost of increased complexity when handling the compressed data with other tools than ADTF. The libjpeg-turbo library uses MMX and SSE2 instructions so you are required to use a CPU that supports these instructions. Valid input video formats are 8 bit grayscale and 24 bit RGB. If you want to compress different formats, please use the Image Conversion Filter from ADTF to change the format.

Table 2.1: Codecs with support on all platforms

2.2 Windows

On Microsoft Windows® the VfW (Video for Windows) API is used additionally for compression and decompression. With the help of this API many VfW codecs may be used. In the following table you can find a summary of codecs that have proven to work well during testing:

Name	Type	Comment
DivX	lossy	Commercial MPEG4 codec. Link: www.divx.com
Xvid	lossy	Open source MPEG4 codec. Link: www.xvid.org
Huffyuv	lossless	Open source codec. Link: http://neuron2.net/www.math.berkeley.edu/benrg/huffyuv.html
MS-MPEG V1	lossy	Microsoft MPEG4 V1 codec. Link: http://www.microsoft.com/windows/windowsmedia/forpros/format/codecdownload.aspx
MS-MPEG V2	lossy	Microsoft MPEG4 V2 codec. Link: http://www.microsoft.com/windows/windowsmedia/forpros/format/codecdownload.aspx

Table 2.2: Codecs with support for Windows only

There is a huge number of other VfW codecs which might work too.

Some VfW codecs may break with an access violation when decoding compressed video (especially data encoded with DivX) by reading past the end of th supplied input buffer. Therefore the *Global Property* ***vc_safe_buffer_size*** can be set to a non-zero value to enable an intermediate buffer that is slightly larger than the original input buffer.

2.3 Linux

On Linx the FFmpeg library is used to support advanced video codecs. Please make sure that you have installed the necessary FFmpeg libraries. Installation instructions for these libraries can be found in the file `Readme.txt` located in the `doc` subdirectory of your ADTF installation. In the following table you can find a summary of FFmpeg codecs that have proven to work well during testing:

Name	Type	Comment
mpeg4	lossy	FFmpeg MPEG4 codec
msmpeg4v2	lossy	FFmpeg implementation of the Microsoft MPEG4 V2 codec. Link: http://www.microsoft.com/windows/windowsmedia/forpros/format/codecdownload.aspx

Table 2.3: Codecs with support for Linux only

There is a large number of other FFmpeg codes which might work too.

2.4 Compatibility

Please note that there is no interoperability between VfW and FFmpeg codecs. Thus *DAT Files* recorded on Windows with VfW codecs can not be played back on Linux and vice versa. Please use the ADTF or JPEG codec in case you require interoperability between Windows and Linux.

This is due to the fact, that the VfW codecs and FFmpeg codecs store the codec parameters in a binary blob, which can not be easily exchanged.

3 ADTF Filters

The *ADTF Video Compression Toolbox* consists of two *Filters* which are described below:

3.1 Image Compression

3.1.1 Description

The *Image Compression Filter* can be used to compress a video stream. The output stream can be stored in a *DAT File*.

3.1.2 Filter-GUID

`adtf.io.image.compress`

3.1.3 Pins

Name	Type	I/O	Description
Input	adtf.type.video	in	This *Input Pin* will receive the input video stream.
Output	adtf.type.video_compressed	out	The compressed video data is transmitted through this *Output Pin*.

Table 3.1: *Pins* of the *Image Compression Filter*

3.1.4 Properties

This *Filter* supports *Dynamic Properties*. After selecting a codec right click on the *Filter* in the *Configuration Editor* and select **Create properties from configuration**. Depending on the codec you will now see additional *Properties*.

Property	Default	Type	Description
Codec	""	String	The codec used for video compression. Hover the mouse pointer above the *Property* to get a list of supported codecs in the *Properties* tool-tip.
ForceKeyframe	tFalse	Bool	Forces each compressed image to be encoded as a keyframe.

Table 3.2: Common *Properties* of the *Image Compression Filter*

VfW Properties

The VfW codecs add the following *Properties* to the *Filter's* list of *Properties*:

Property	Default	Type	Description
CodecConfigurationFilename	""	String	Path to a codec configuration file to use. See chapter 4 Configuration of VfW codecs on how to create such a file.

Table 3.3: VfW-related *Properties* of the *Image Compression Filter*

FFmpeg Properties

The FFmpeg codecs add the following *Properties* to the *Filter's* list of *Properties*.

Property	Default	Type	Description
Bitrate	800000	int	This is the bitrate aimed at, which controls the compression quality. Please note that this bitrate is based on a framerate of 25. If the input framerate differs from 25 then the bitrate will differ the same ratio.

Table 3.4: FFmpeg-related *Properties* of the *Image Compression Filter*

3.2 Image Decompression

3.2.1 Description

The *Image Decompression Filter* is the counterpart of the *Image Compression Filter* and can be used to decompress a compressed video stream.

3.2.2 Filter-GUID

`adtf.io.image.decompress`

3.2.3 Pins

Name	Type	I/O	Description
Input	adtf.type.video_compressed	in	This *Input Pin* will receive the compressed input video stream.
Output	adtf.type.video	out	The video data is transmitted through this *Output Pin*.

Table 3.5: *Pins* of the *Image Decompression Filter*

3.2.4 Properties

This *Filter* has no *Properties*.

4 Configuration of VfW codecs

Some VfW codecs can be configured by the user. To offer this feature in ADTF the tool *VfW Codec Configuration Utility* may be used. It will be installed only together with the Microsoft Windows® version of the toolbox. On start the following dialog will be displayed:

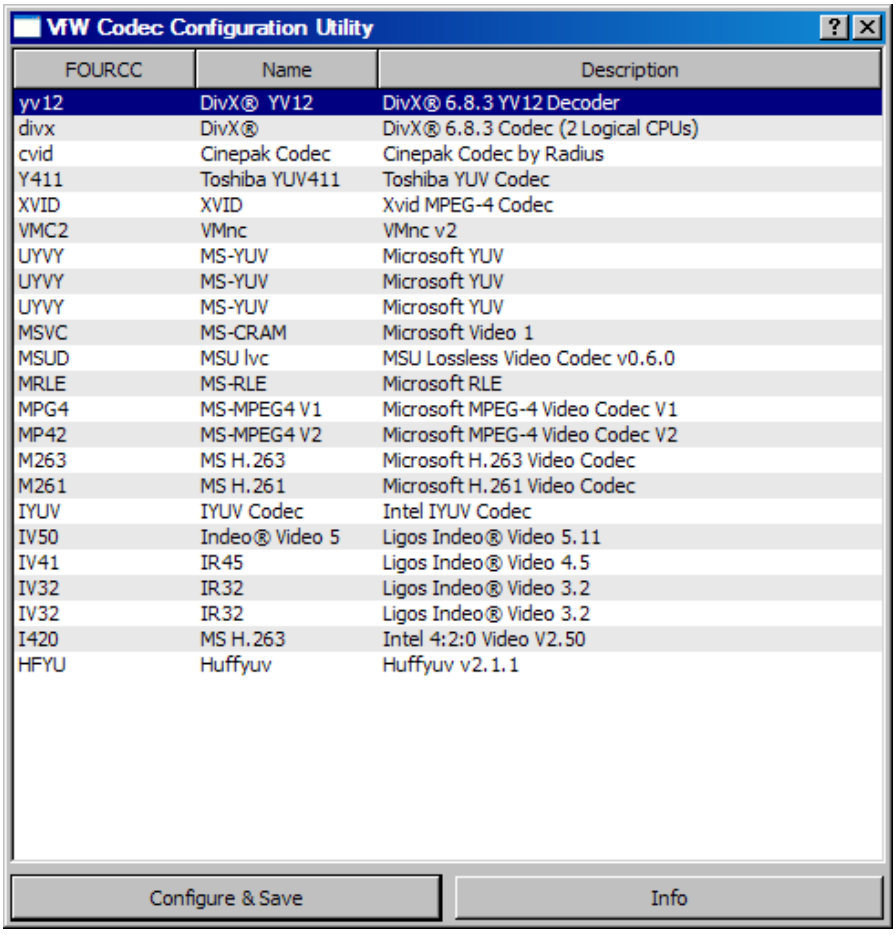

Figure 4.1: VfW Codec Configuration Utility

This dialog displays a list of all installed VfW codecs. In the column FOURCC the codec identifier is displayed.

When the button **Info** is pressed the version information dialog of the selected codec will be displayed. Not all VfW codecs support this feature.

When the button **Configure & Save** is pressed the configuration dialog of the selected codec will be displayed. The settings dialog is implemented by the codec itself and is not issue of this document. After closing the settings dialog the settings can be stored in a XML file.

The generated XML file can be used by the ADTF *Filters* which provide the *Property* ***CodecConfigFileName***. The configuration stored in the XML file will be restored.

- Not all VfW video codecs allow saving configuration information.
- It can not be guaranteed that stored configurations are usable with different versions of the VfW codec or on different computers at all. This has to be tested by the user himself.

5 Export and import of compressed video streams

The *Video Compression Support Service* provides *Plugins* for the ADTF *DAT Exporter*. If you use the command-line *DAT Exporter* be sure to specify the *Services* from this toolbox with the help of the `-service` switch of the ADTF *DAT Exporter*.

5.1 Export

You can export compressed video streams in the same manner as you do with uncompressed video streams, just select the *Compressed Video Sink* in the ADTF *DAT Exporter* GUI or use the command line.

Example for using command line:

```
C:\ADTF\bin\adtf_datexporter.exe -export xvid_compressed -sinkid
adtf.export.compressed_video -output C:\Files\new_video.avi
-property Codec=XVID -service C:\ADTF\addons\compression-toolbox-trunk\bin\
video_compression_support.srv D:\Files\xvid_clock.dat
```

5.2 Import

To import a video file as a compressed video stream into a new *DAT File* select the file to import with the help of the **Add source file** button in the **Create New Dat File** dialog and select the ADTF *Compressed Video Source*.

On the command line use following source ids:
Windwos:

- `adtf.import.compressed_video`.
- `adtf.import.vfw`

Linux:

- `adtf.import.ffmpeg`

Currently you can only select the codec which should be used for importing the file, but cannot set any additional codec *Properties* via the GUI (this is a limitation to the GUI). If you need to set additional *Properties* use the command-line *DAT Exporter* where you can specify additional *Properties* with the switch `-property name=value`, e.g:

```
C:\ADTF\bin\adtf_datexporter.exe  -import C:\Files\
existing_video.avi -sourceid adtf.import.compressed_video
-property Codec=XVID -service C:\ADTF\bin\addons\
adtf-compression-toolbox-trunk\bin\video_compression_support.srv
C:\Files\new_dat_file.dat
```

6 FAQ

- Problem: When the codec Xvid is used for compression ADTF is blocked and a gray window is shown on the screen.

  Solution: In the configuration of the Xvid codec the option “Display encoding status” has to be switched off.

- Problem: When the codec DivX is used for compression the *Filter* fails to create any output data.

  Solution: In the configuration of the DivX codec the certification profile “Profile 1080HD” has to be selected. By this profile the limitations of image formats are defined.

- Problem: It is not possible to compress 16 bit grayscale images using VfW codecs.

  Solution: Most VfW codecs are only capable to compress images with 24 bit RGB or HYU format. Therefore the input video has to be transformed beforehand by using the ADTF *Image Transform Filter*

- Problem: When the codec HFYU is used and the option “RGB Compression method” is set to “<– Convert to YUV” the decompressed video is distorted.

  Solution: In the configuration of the codec HFYU the option “Always suggest RGB for output format” has to be activated.

- Problem: The settings of the VfW codec HFYU can not be saved by the *VfW Codec Configuration Utility*.

  Solution: On each PC the settings have to be made by hand. They will be stored in the registry by the codec.

- Problem: If video recording with compression is used and the first received *MediaSample* is not a full image, just a black picture is shown up to the next frame.

  Explanation: The image compressor would have to react on a recorder event from *ADTF Harddisk Player*, eg. *StartRecording* or *RecFileOpen* and create a key frame. This does not cover all use cases. If the recorder is used with a queue, it is possible that all *MediaSamples* at the queue do

not have the according key frames. Only after saving the queue the key frame would be included into the file.

Solution: Use a codec which doesn't need key frames! E.g. the *ADTF Codec*.

List of Figures

List of Tables

Index