MAWT - Movie Automation With Tcl

poSoft

A Work-In-Progress report
by Paul Obermeier

MAWT = FFmpeg + tcl tk

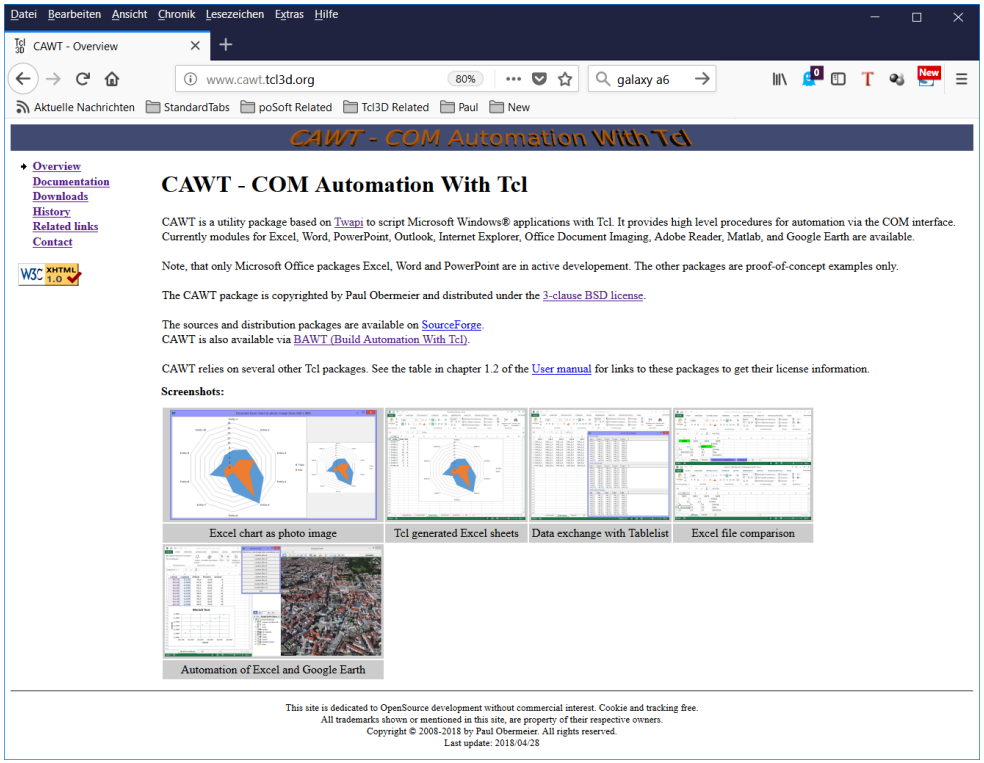
- **Introduction to Automation With Tcl**
- **MAWT Architecture**
- **MAWT Internals**
- **MAWT Status**
- **Summary**

Introduction - Automation With Tcl

poSoft

Episode 1: **CAWT** – COM Automation With Tcl

- Introduced at EuroTcl 2012. Currently at version 2.4.2.
- High-level Tcl interface for scripting Windows applications having a COM interface.
- Available at `http://www.cawt.tcl3d.org`
- Also available as part of BAWT-Tcl.

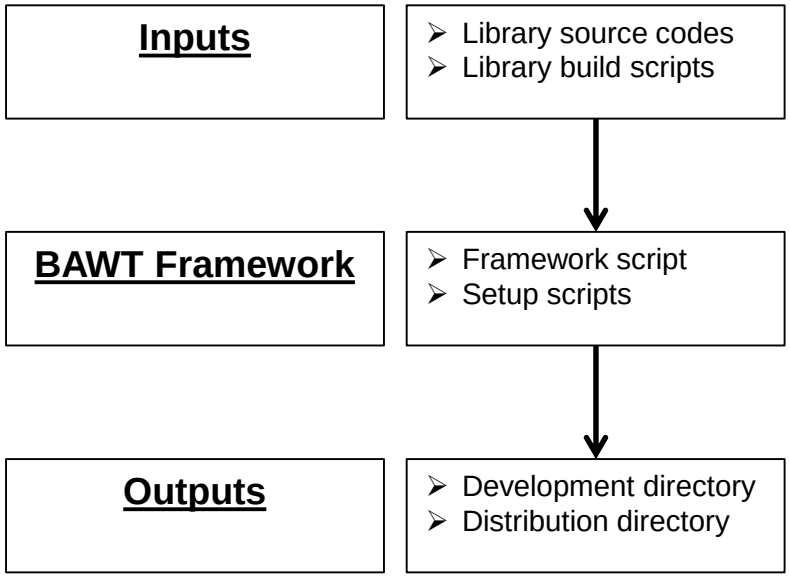


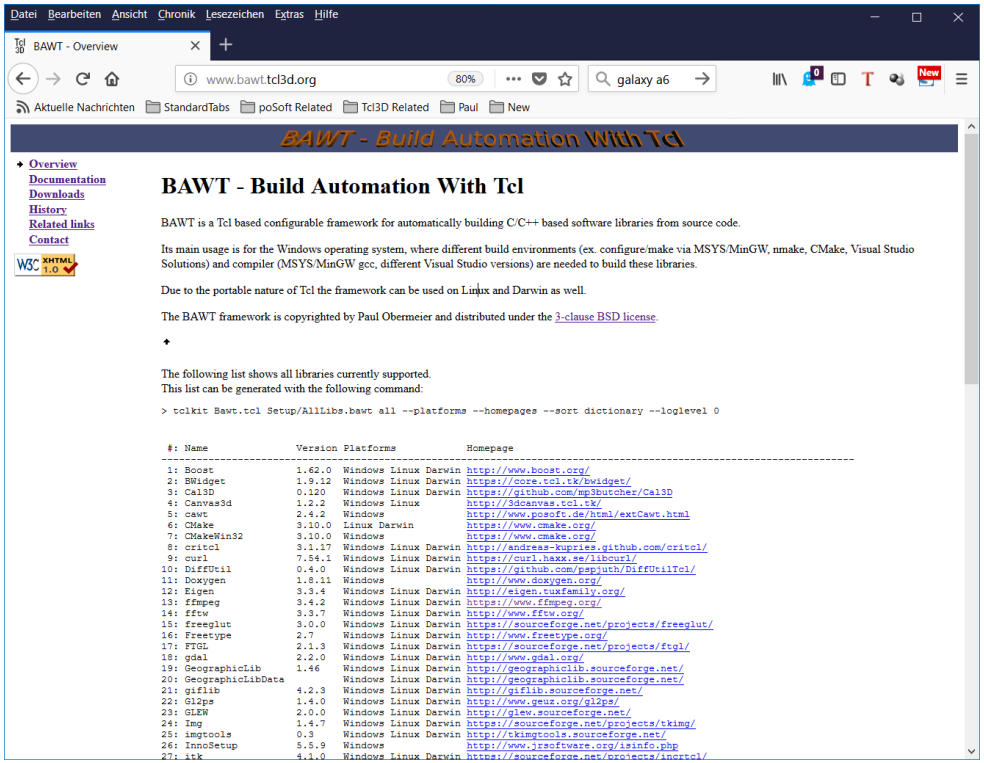
Introduction - Automation With Tcl

poSoft

Episode 2: **BAWT** – Build Automation With Tcl

- Introduced at EuroTcl 2016. Currently at version 0.8.0.
- Tcl based framework for automatically building C/C++ based libraries from source code.
- Available at `http://www.bawt.tcl3d.org`



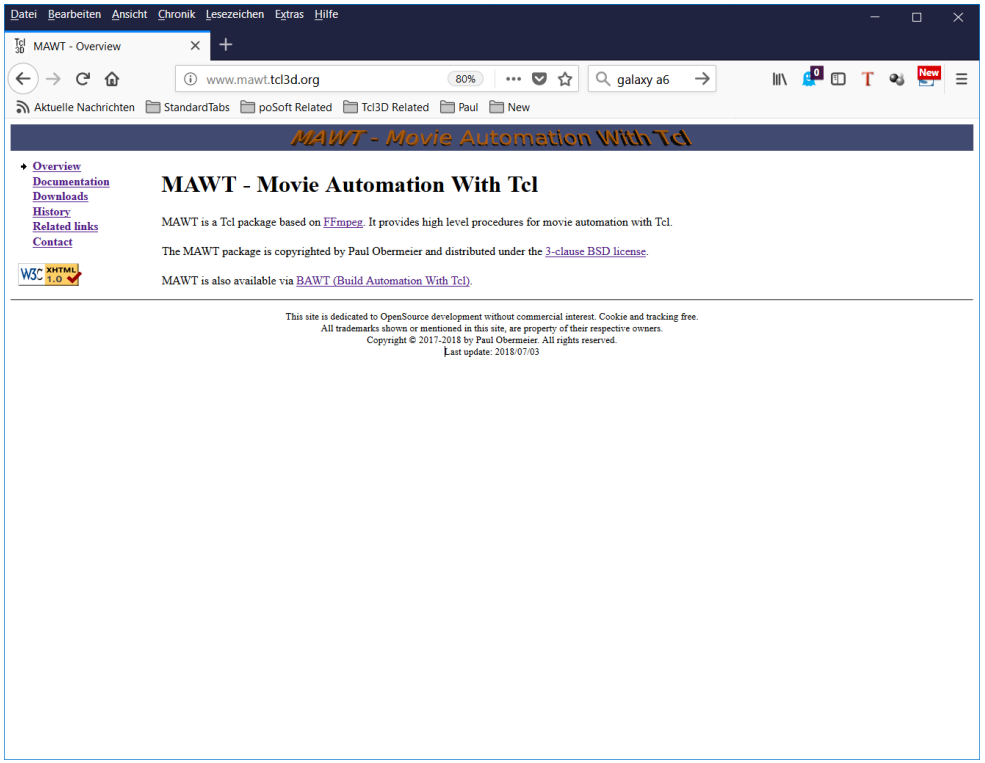


Introduction - Automation With Tcl

poSoft

Episode 3: **MAWT** – Movie Automation With Tcl

- Introduced at EuroTcl 2018. Currently at version 0.1.0.
- High-level Tcl interface offering some of the video functionality of FFmpeg.
- Available at `http://www.mawt.tcl3d.org`
- Also available as part of BAWT-Tcl.



Introduction - FFmpeg

poSoft

- A complete, cross-platform solution to record, convert and stream audio and video.
- It is programmed in the ISO C90 language with a few additional features from ISO C99.
- Available at `http://www.ffmpeg.org`

FFMPEG consists of several libraries:

libavutil contains functions for simplifying programming (ex. mathematics routines, core multimedia utilities, …).

libavcodec contains decoders and encoders for audio/video codecs.

libavformat contains demuxers and muxers for multimedia container formats.

libavdevice contains input and output devices for access to many common multimedia input/output software frameworks.

libavfilter contains media filters.

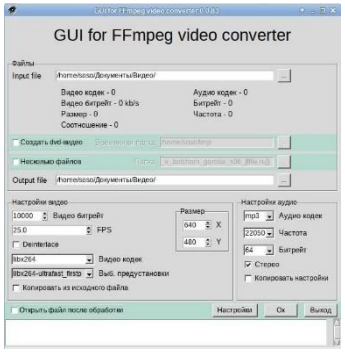
libswscale performs highly optimized image scaling and color space/pixel format conversion operations.

libswresample performs highly optimized audio resampling, rematrixing and sample format conversion operations.

Introduction – Existing Tcl Video Packages

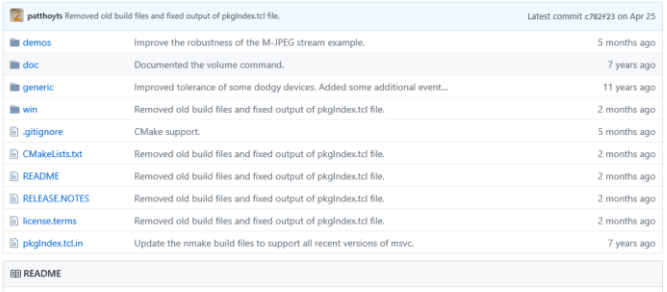
- **tkffmpeg** Graphical user interface for FFmpeg batch program.

  https://sourceforge.net/projects/tkffmpeg/



- **tkvideo** DirectShow binding. Windows only.

  http://www.patthoyts.tk/tkvideo/

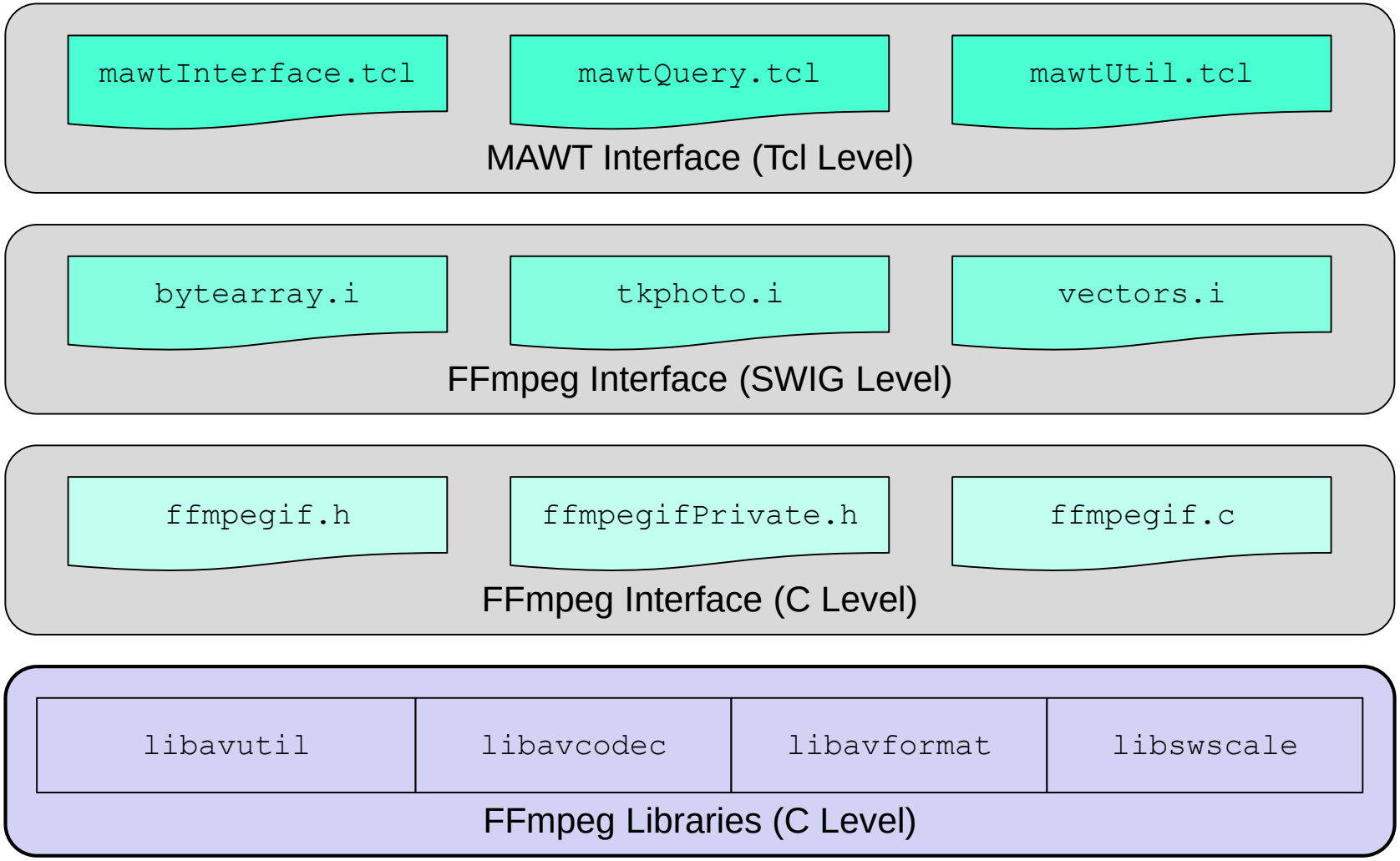


- **QuicktimeTcl** Binding for Quicktime.

  http://quicktimetcl.sourceforge.net/

MAWT Architecture – Layer Overview

MAWT is realized as a 3 layer architecture.



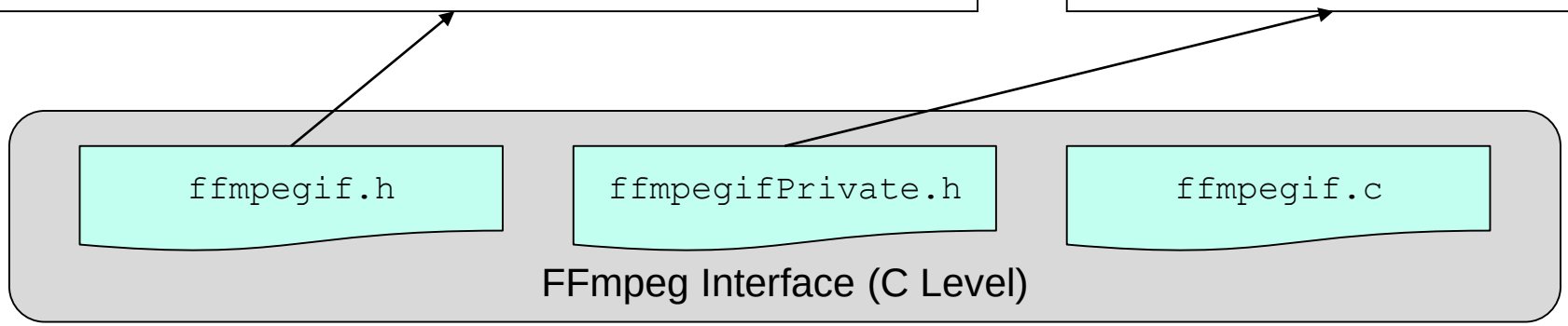
MAWT Architecture – C Level

poSoft

The FFmpeg C interface defines high level functions for handling video data

```
typedef struct videoStruct *VideoId;

VideoId VideoOpen  (const char *fileName);
void    VideoClose (VideoId videoId);
Bool VideoGetNextImage (VideoId videoId, void *img, Bool flip);

VideoId VideoCreate(const char * fileName, int width, int height);
Bool    VideoFinish(VideoId videoId);
Bool    VideoWriteNextImage(VideoId videoId, void *img, Bool flip);

int     VideoStart (VideoId videoId, int width, int height);
void    VideoStop  (VideoId videoId);

int     VideoGetNumFrames  (VideoId videoId);
double  VideoGetFramerate  (VideoId videoId);
int     VideoGetWidth      (VideoId videoId);
int     VideoGetHeight     (VideoId videoId);

const char *GetErrorMessage (void);
const char *GetFfmpegVersion (void);
```

```
typedef struct videoStruct {
    struct SwsContext *swsCtx;
    AVFormatContext *formatCtx;
    AVCodecContext *codecCtx;
    AVOutputFormat * outFmt;
    AVFrame *rgbFrame;
    AVFrame *yuvFrame;
    AVStream * stream;
    AVPacket packet;
    int videoStream;
    int audioStream;
    char name[100];
    int numFrames;
    int started;
    int curFrame;
    int width, height;
    int outWidth, outHeight;
    unsigned int numTotalFrames;
    double frameRate;
    FILE *fp;
} VideoStruct;
```



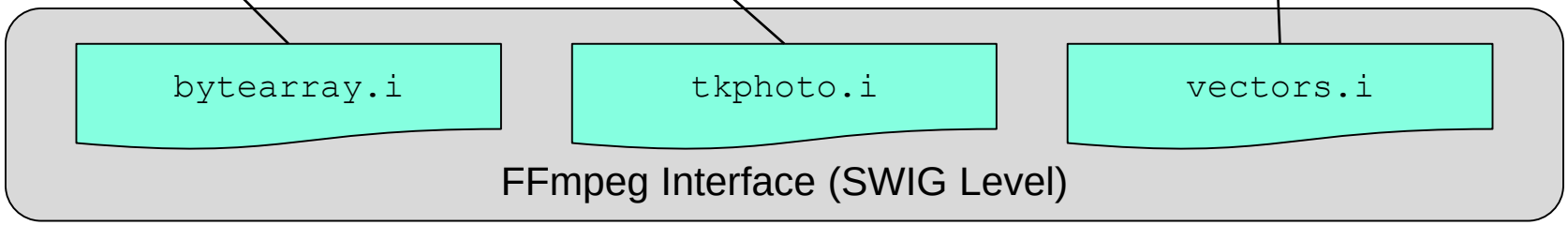
MAWT Architecture – SWIG Level

The FFmpeg SWIG interface implements low level C functions for handling vector data and transferring vector data to/from byte arrays and photo images.

```
int VectorFromByteArray (
    Tcl_Interp *interp,
    const unsigned char *src,
    void *dest,
    int numBytes,
    int srcOff,
    int destOff
);

int VectorToByteArray (
    Tcl_Interp *interp,
    const void *src,
    unsigned char *dest,
    int numBytes,
    int srcOff,
    int destOff
);
```

```
int VectorFromPhoto (
    Tcl_Interp *interp,
    char *imgName,
    void *imgVec,
    int numChans,
    float scale,
    float offset
);

int VectorToPhoto (
    Tcl_Interp *interp,
    void *imgVec,
    const char *photoName,
    int width,
    int height,
    int numChans,
    int linesAreBottomUp);
```

```
%define %baseTypeVector(TYPE,NAME)
…
static TYPE *new_##NAME(int nelements)
{
  return (TYPE *)calloc(nelements,sizeof(TYPE));
}

static void delete_##NAME(TYPE *ary)
{
  free(ary);
}
…
%enddef

// Generate vector functions
// for the following types.
%baseTypeVector(unsigned char, UByte)
%baseTypeVector(char, Byte)
```



MAWT Architecture – Tcl Level

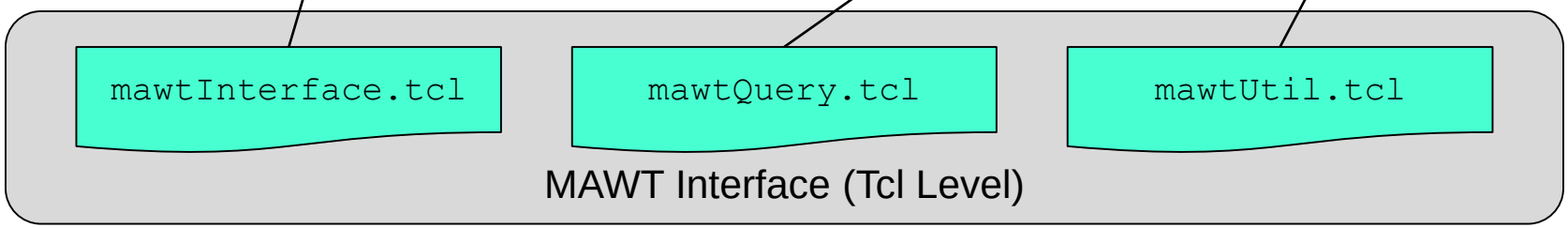
The MAWT Tcl interface defines the TclOO classes **Video** and **Vector**.

It also adds query and utility procedures.

```
 oo::class create Video {
  variable mVideoId

  method Open { fileName } {
    set mVideoId [ffmpegif::VideoOpen $fileName]
  }
  method GetNumFrames {} {
    return [ffmpegif::VideoGetNumFrames $mVideoId]
  }
}

oo::class create Vector {
  constructor { numBytes } {
    set mVec [ffmpegif::new_UByte $numBytes]
  }
  method FromPhoto
  method ToPhoto
}
```

```
proc GetFfmpegVersion {}
```

```
proc VideoExport {
  movieFile
  imgFilePattern
  args
}

proc VideoImport {
  imgFilePattern
  movieFile
  args
}
```



MAWT Internals - Dependencies

poSoft

Dependencies to build MAWT:

- `CMake`
- `SWIG`
- `FFmpeg`
- `Tcl/Tk`

Dependencies to run MAWT:

- `Tcl/Tk`
- `FFmpeg`
- *`img::raw (optional)`*

MAWT Internals – Transfer Mode Put

- Create photo image, video and vector object.
- Attach photo image to label.
- Get next video frame.
- Create bytearray from vector data. ***1 copy***
- Update photo image with **put** command using the bytearray data. ***1 copy***

ModePut.tcl (C:\poSoft\tcl3d-Sam...rtraege\EuroTcl2018-Mawt) - GVIM1

Datei Editieren Werkzeuge Syntax Puffer Ansicht Hilfe

```
# Initialization
label .l
image create photo VideoFrame
.l configure -image VideoFrame

set videoObj [mawt Video new $movieFile]
set vectorObj [mawt Vector new $numBytes]

# Display loop
while { true } {
  $videoObj GetNextImage [$vectorObj Get]
  set imgBinVar [$vectorObj ToByteArray [expr {3 * $tx * $ty}]]
  VideoFrame put $imgBinVar -format \
              "raw -nomap 1 -width $tx -height $ty -nchan 3 -useheader 0"
  update
}
                                                    9,1           Alles
```

MAWT Internals – Transfer Mode Create

- Create video and vector object.
- Get next video frame.
- Create bytearray from vector data. ***1 copy***
- Create photo image with **image create** command using the bytearray data. ***1 copy***
- Attach photo image to label.

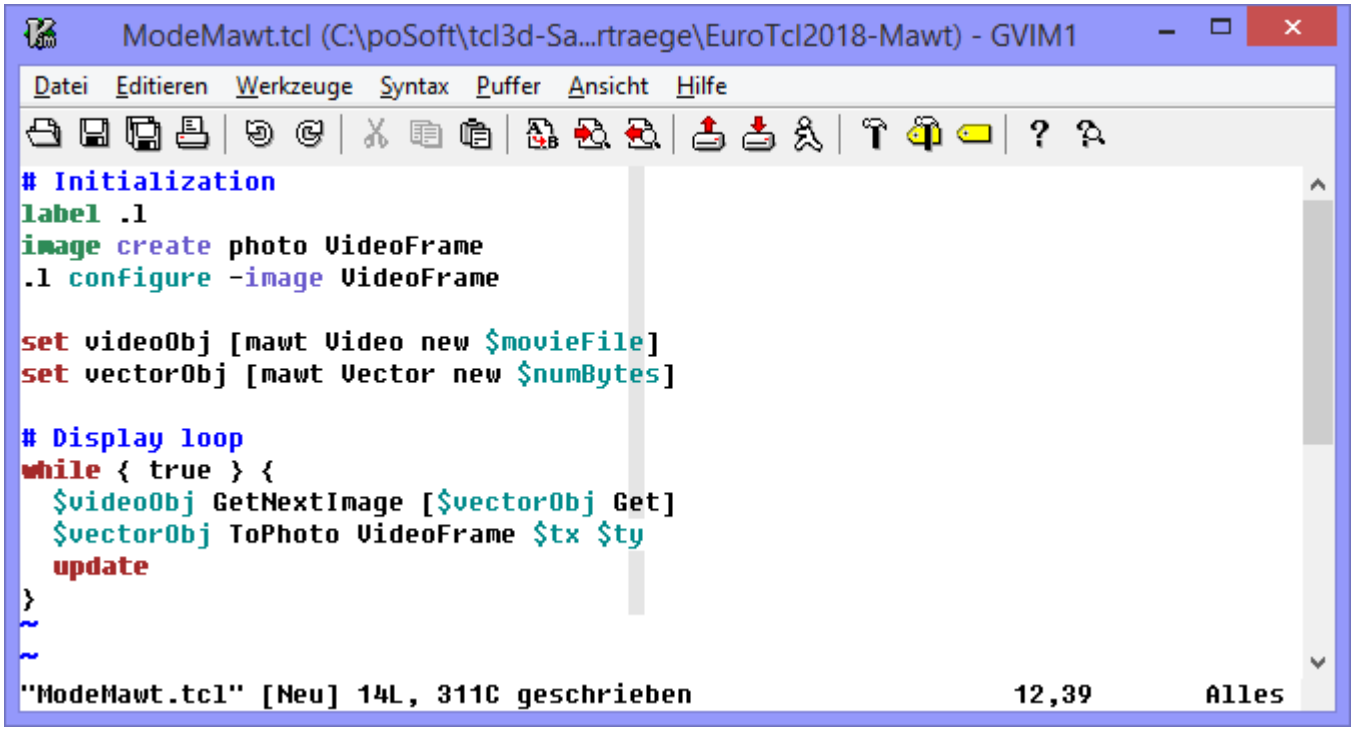
ModeCreate.tcl (C:\poSoft\tcl3d-...rtraege\EuroTcl2018-Mawt) - GVIM1

Datei Editieren Werkzeuge Syntax Puffer Ansicht Hilfe

```
# Initialization
label .l

set videoObj [mawt Video new $movieFile]
set vectorObj [mawt Vector new $numBytes]

# Display loop
while { true } {
  $videoObj GetNextImage [$vectorObj Get]
  set imgBinVar [$vectorObj ToByteArray [expr {3 * $tx * $ty}]]
  catch { image delete VideoFrame }
  image create photo VideoFrame -data $imgBinVar -format \
        "raw -nomap 1 -width $tx -height $ty -nchan 3 -useheader 0"
  .l configure -image VideoFrame
  update
}
"ModeCreate.tcl" [Neu] 16L, 472C geschrieben                13,2          Alles
```

MAWT Internals – Transfer Mode Mawt

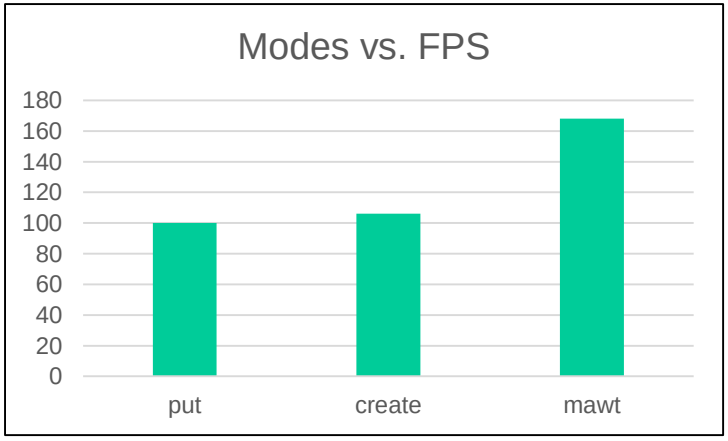
- Create photo image, video and vector object.
- Attach photo image to label.
- Get next video frame.
- Update photo image with **ToPhoto** command of Vector class. ***1 copy***



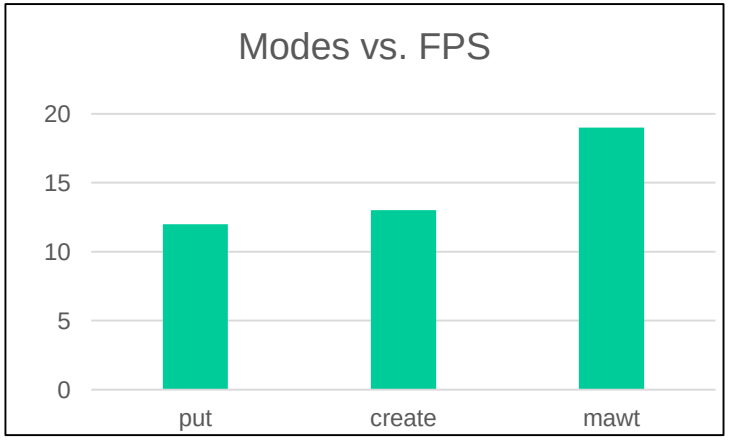
MAWT Status – Performance Measurements

Performance measurements done with script `speedTest.tcl`

Comparison of transfer modes		
Mode	Size	FPS
put	640x464	100
create	640x464	106
mawt	640x464	168



Comparison of transfer modes		
Mode	Size	FPS
put	1920x1080	12
create	1920x1080	13
mawt	1920x1080	19

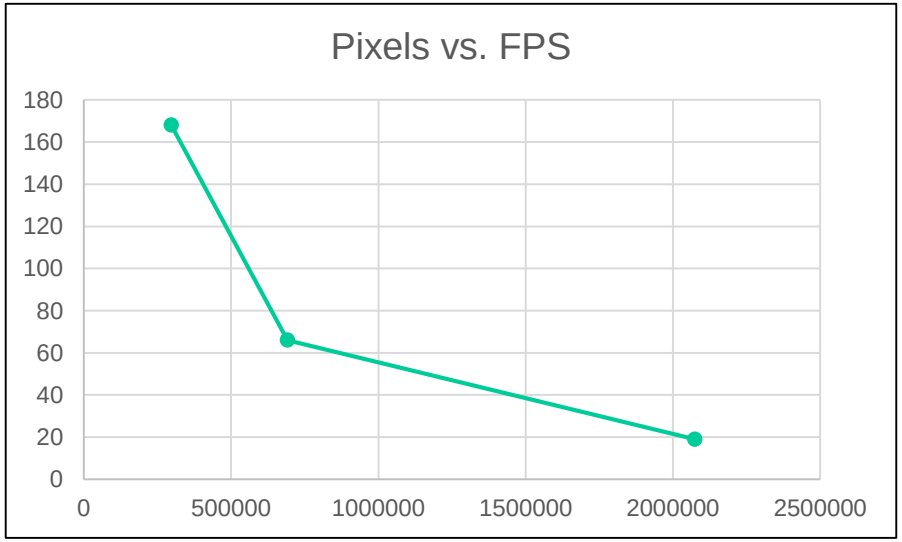


CPU: Core i7-4700HQ 2.4GHz

MAWT Status – Performance Measurements

Performance measurements done with script `speedTest.tcl`

Comparison of video sizes			
Mode	Size	Pixels	Measurement
mawt	640x464	296,960	Faster than needed: 6.70 times (168 fps)
mawt	960x720	691,200	Faster than needed: 2.66 times (66 fps)
mawt	1920x1080	2,073,600	Faster than needed: 0.62 times (19 fps)

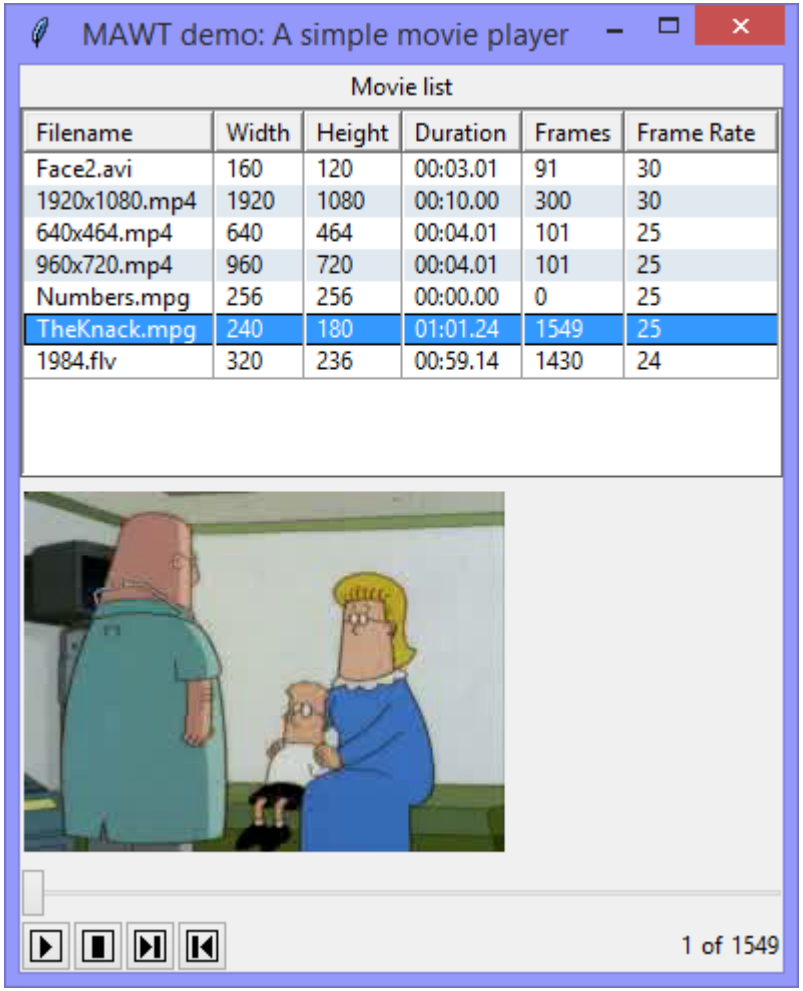


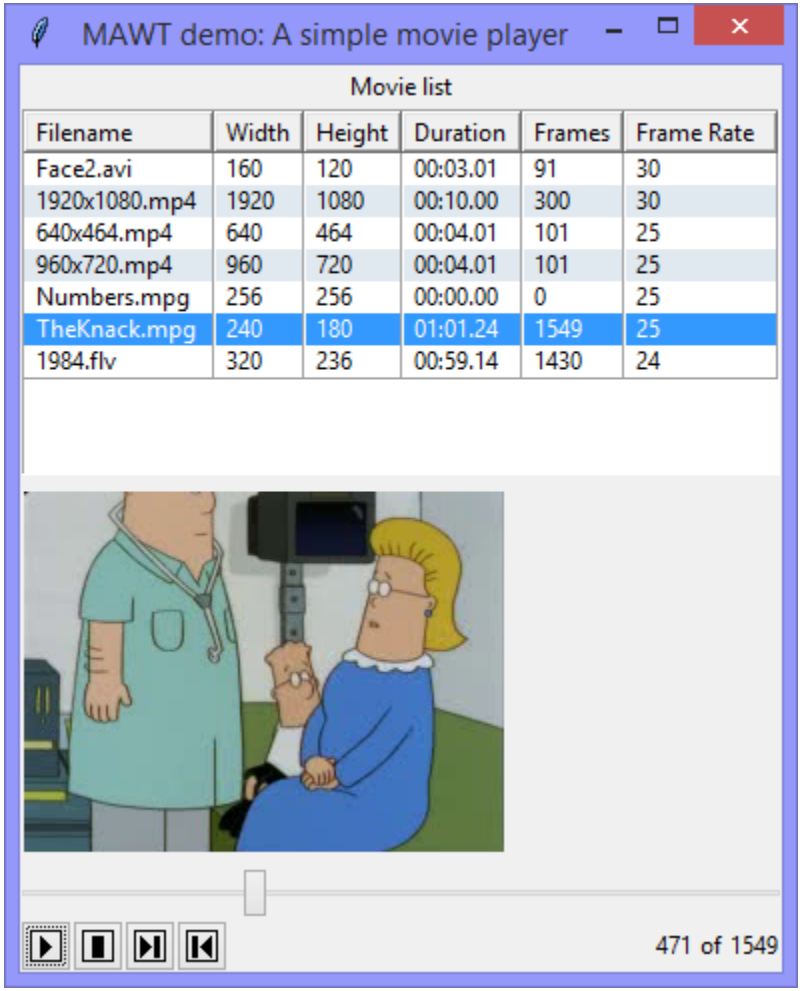
CPU: Core i7-4700HQ 2.4GHz

MAWT Status – Demo and Test Programs

poSoft

MAWT Demo: Simple movie player `movieplayer.tcl`

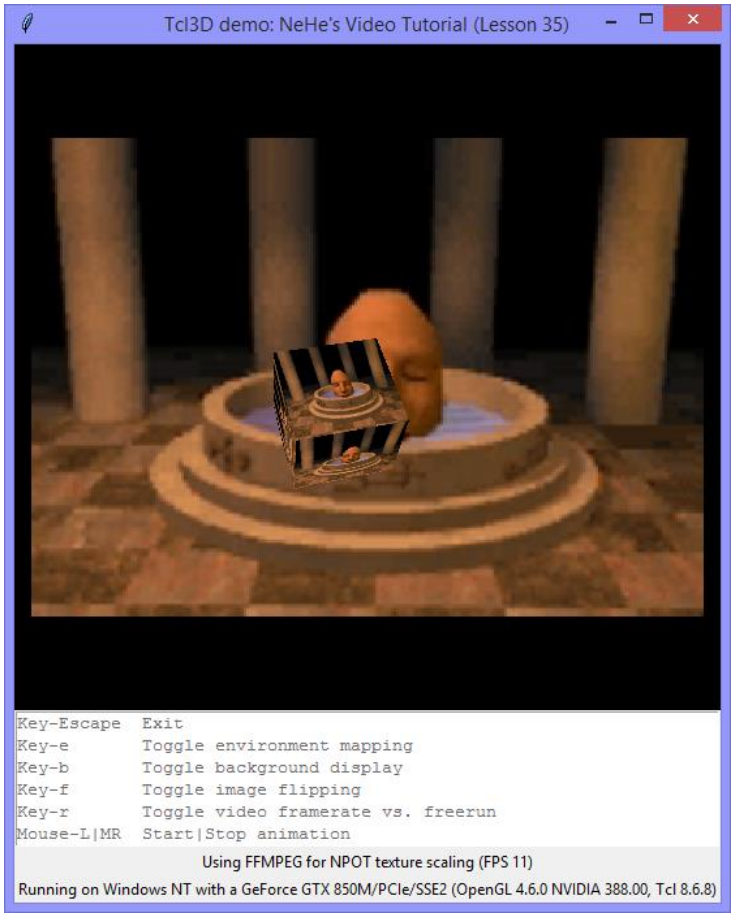




MAWT Status – Demo and Test Programs

poSoft

MAWT Demo: Tcl3D with animated textures `NeHe-Lesson35.tcl`



MAWT Status – Demo and Test Programs

poSoft

MAWT Demo: Test scripts for procs `VideoImport` and `VideoExport`

```
> tclsh VideoExport.tcl
FFmpeg version    : 3.4.2
Movie file        : Data/Face2.avi
Image file pattern: TestOut/Face.%03d.png
Number of images  : 51
Time needed       : 0.16 seconds
```

```
> tclsh VideoImport.tcl
FFmpeg version    : 3.4.2
Image file pattern: TestOut/Face.%03d.png
Movie file        : TestOut/Face2.mpg
Number of images  : 51
Time needed       : 0.10 seconds
```

Summary

poSoft

Current status

- ✓ Architecture and build structure defined.
- ✓ Video display with Tcl/Tk and FFmpeg is possible.
- ✓ Exporting images from a video is possible.
- ✓ Generating a video from images is possible.

Future work

- Improve display performance.
  - Implement zero-copy transfer mode.
  - Improve FFmpeg compilation in BAWT.
- Improve support of video formats.
- Improve FFmpeg interface.
- *Add audio support.*

MAWT Resources	
MAWT Sources (Homepage of MAWT)	http://www.mawt.tcl3d.org
MAWT Binaries (Part of BAWT-Tcl)	http://www.bawt.tcl3d.org