MKQT Manual

2010-10-23

Laborpraktikum

Otto-von-Guericke Universität Magdeburg

Department of Computer Science

Ronny Wegener

Table of Contents

1. Introduction

The aim of this project is to present 3D objects, created by MeVisLab, interactively in web-pages. There are already some plug-ins and frameworks available to render polygonal and volume models directly in a web-browser, but these requires a conversion of the model data and the installation of uncommon browser plug-ins. To avoid these requirements another solution is used. Instead of converting the model directly, a set of pre-rendered images from different viewpoints (pan/tilt) will be created. These images can be used to emulate a 3D behavior by showing the image corresponding to the users viewpoint. An interface is needed to capture the users interaction and show the correct image. QuickTime already offers such a pseudo 3D file format based of pre-rendered images and as a common plug-in for different browsers, the decision was made to presenting MeVisLab's pre-rendered images by QuickTime.

Some proprietary and free authoring tools are available to convert images to the QuickTime VR container, but most of them are focused on panoramas instead of objects. Those which supports objects are over flooded with features and cannot be integrated into MeVisLab for automatic image conversion. The task is to create a simple command line tool that can be used in MeVisLab for batch conversion of images into the QuickTime VR file format.

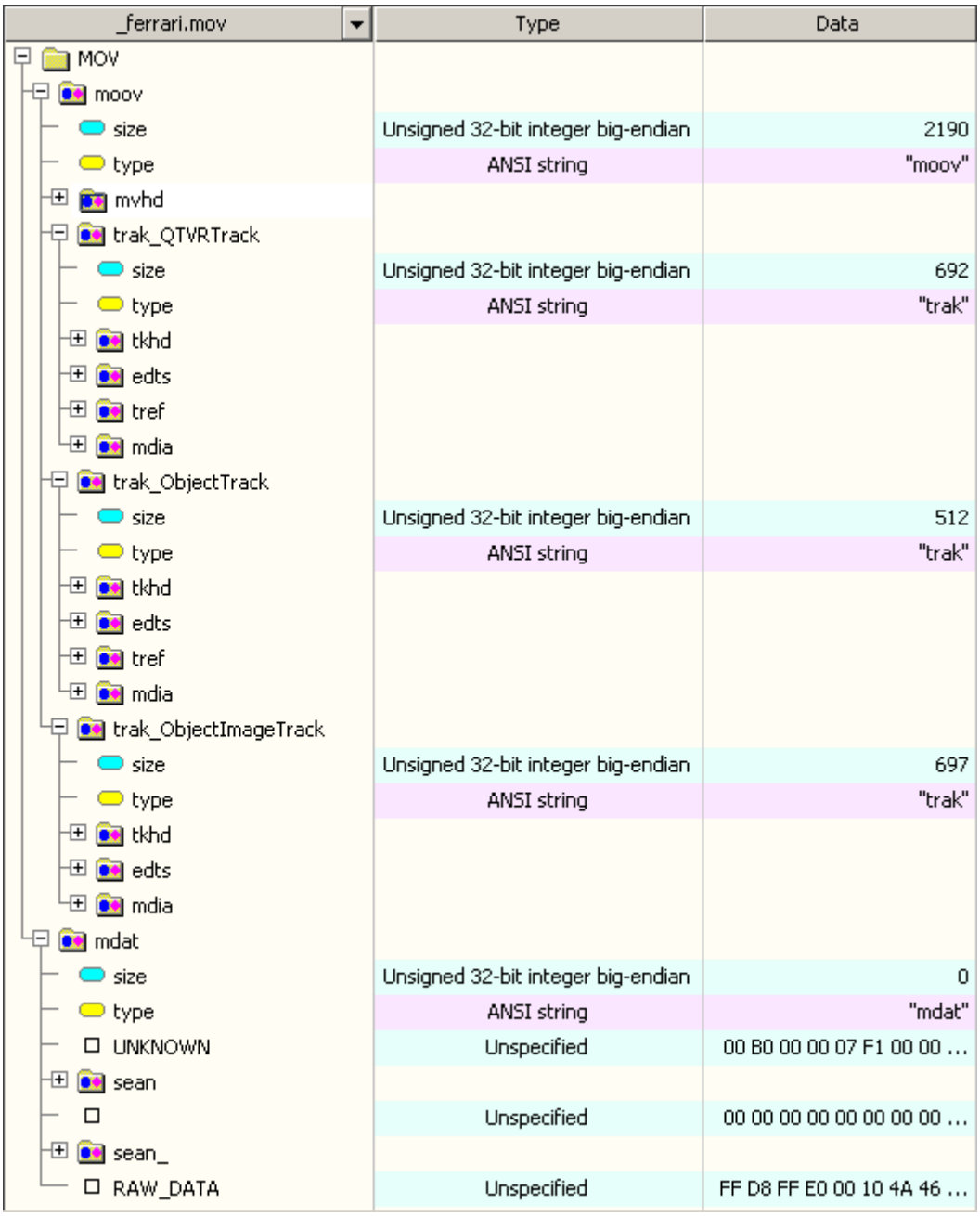
The first part of this document gives an overview of the development process of mkqt, a tool for converting images to QuickTime in a simple batch style manner. The second part is more user oriented and will guide through the creation and requirements of the images and how these images can be converted into a QuickTime VR file using the developed tool.

2. MKQT Development

The following chapters will give a brief overview of the development process of mkqt, a simple command line utility that converts images into a QuickTime VR movie.

2.1. QuickTime Basics

The QuickTime SDK for windows, which is currently outdated and bad documented, will not be used, so it is necessary to take a basic look into the QuickTime VR file format. QuickTime is a container format for multimedia data in various compressions and additional data that holds the description of these multimedia data. A QuickTime movie is structured in units that are called atoms. Each atom contains the size of the atom and a type identifier. Depending on the atom type it is followed by structured raw data or other atoms inside this atom.



_ferrari.mov	Type	Data
MOV		
moov		
size	Unsigned 32-bit integer big-endian	2190
type	ANSI string	"moov"
mvhd		
trak_QTVRTrack		
size	Unsigned 32-bit integer big-endian	692
type	ANSI string	"trak"
tkhd		
edts		
tref		
mdia		
trak_ObjectTrack		
size	Unsigned 32-bit integer big-endian	512
type	ANSI string	"trak"
tkhd		
edts		
tref		
mdia		
trak_ObjectImageTrack		
size	Unsigned 32-bit integer big-endian	697
type	ANSI string	"trak"
tkhd		
edts		
mdia		
mdat		
size	Unsigned 32-bit integer big-endian	0
type	ANSI string	"mdat"
UNKNOWN	Unspecified	00 B0 00 00 07 F1 00 00 ...
sean		
	Unspecified	00 00 00 00 00 00 00 00 ...
sean_		
RAW_DATA	Unspecified	FF D8 FF E0 00 10 4A 46 ...

Image 01: selection of the vr object movie structure viewed with HexEdit Free 3.0

The media is organized in tracks, these are virtually divided into multiple chunks and each chunk contains the samples. Also an own time coordinate system is implemented, which uses ticks as unit. The number of ticks that a second hold is defined in the movie file. The VR file format contains many atoms and a lot of parameters. More detailed information can be found in the QuickTime File Format 2001 specification by Apple Computer, Inc.

2.2. Simplifying the QuickTime VR Structure

As described in the previous chapter, there are a lot of different atoms, but with respect to the task only the atom structure of a QuickTime VR Object movie is required. The QuickTime atom structure for VR Object movies is static so it is not necessary to take care about swapping atoms. To figure out which atoms are basically required and which can be dismissed an existing sample VR Object movie was examined using a hex editor with structure viewing support. At the same time the results were compared and verified with Apple's QuickTime file format specification sheet. After deep research the basic atom structure, required for an implementation, was fully explored.

The QuickTime VR Object movie consists of two major atoms, the moov atom that contains all the description data and the mdat atom that contains the raw data. The description atom contains a header atom and three trak atoms (one qtvr track, one object track and one object image track). In each track an absolute file offset in bytes can be found, which links to the raw data in the mdat atom. A closer look into the atoms will not be performed at this point, detailed information can be found in the corresponding file format specifications offered by Apple Computer Inc.

2.3. MKQT Design

The minimized structure for the VR object movie should be the base and is directly embedded into the source of the developed tool. When a conversion process is initialized the first step is to instantiate the embedded base structure. The second step is to customize the instantiated base structure depending on the input data. In this step some run-time parameters will be generated and changed in the base structure, also the input data will be streamed into the base structure. At least the final structure will be written to a file.

The customization step is the most important point of interest. The following paragraph will give a short description how the input data will be

streamed into the structure. To keep things simple, the input data is reduced to an image matrix, which can be represented as a list of images and a number defining the rows of the resulting matrix. All images will be re-compressed using JPEG compression with a quality rating of 85 and streamed into the mdat atom. The file offset for each image is stored in the corresponding trak atom. The width and height information for all description atoms is acquired directly from the image files. There is an default view position defined by tilt angle and pan angle. The pan angle is always set to 0, but the tilt angle is calculated as the centered tilt of all tilt angles, that means if you have images taken from 3 different tilt angles [20°, 0°, -20°], the default tilt is [0°]. The time coordinate system is also affected, the total number of images is directly used as number of ticks per second, where each image is a single tick so overall length in ticks is exactly the number of images and the overall length in seconds is always 1.

3. MKQT Application

The following chapters are focused for general users. All steps that are necessary to prepare the images and create a QuickTime VR Object movie from a set of images will be explained in the next pages.

3.1. Image Creation and Requirements

All images that should be used for the conversion must be in the same directory. Images that should not be included have to be removed from this directory. Supported image formats are png, bmp, jpg, gif, tif, tga and xpm all without transparency. All images must have the same dimension (width x height). The images must be ordered by name, it is recommend to use a 3 digit code followed by a dot and the file extension (i.e. "003.png").

The correct image ordering starts at the top most tilt angle and step counterclockwise through all pan angles belonging to this tilt, this is related to a left oriented rotation around the object. When finished the next tilt angle will be processed by sweeping through all belonging pan angles again. These steps

are repeated until the last tilt angle is finished. The following image shows an example file naming to order images of a skull, but instead of different tilt angles the picture contains different x-ray levels.

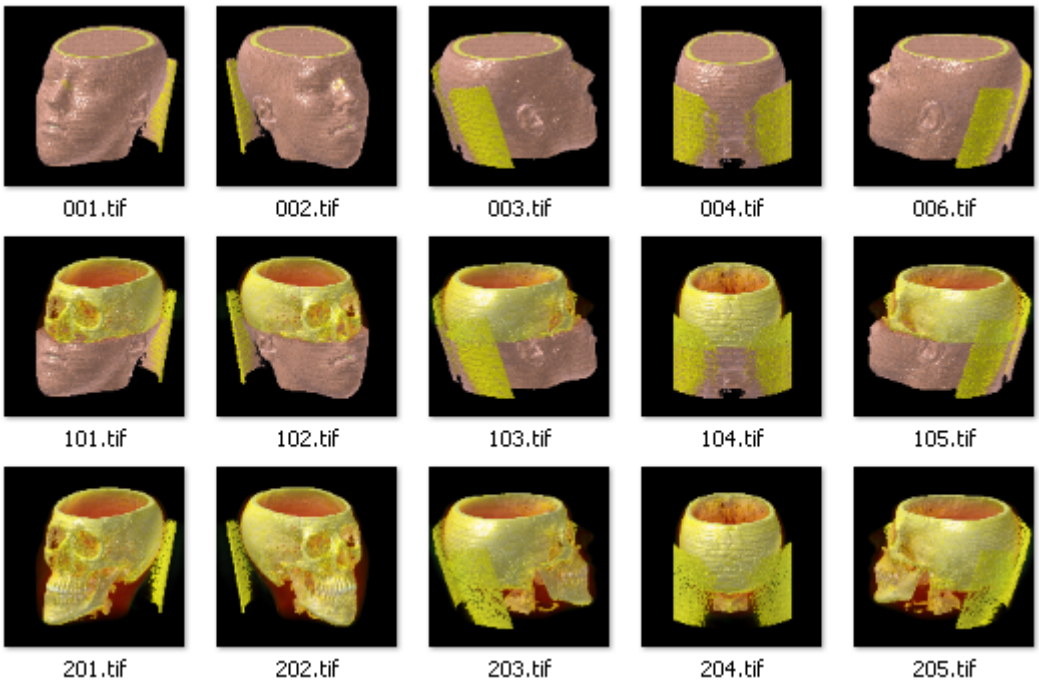


Image 02: file naming -> image order

3.2. Converting Images with MKQT

After all images are prepared and ordered correctly, mkqt can be used to create the QuickTime VR Object movie. The command is

mkqt.exe path tiltcount

where **path** is the absolute or relative path to the directory that contains the images and **tiltcount** is the total number of tilt angles. The number of images in the directory must be a multiple of tiltcount, because those values are used to create the two dimensional image matrix. If there are for example 15 images in the directory and these images are taken from 3 different tilt angles then call **mkqt.exe directory 3** and the internal used image matrix has 3 rows and 5 columns.

3.3. MKQT GUI Mode

Beside the command line version there is the possibility to start the application in GUI mode which gives access to advanced features. The GUI mode will be started in the following situations:

- mkqt.exe is executed without command line arguments
- mkqt.exe has less or more than two command line arguments
- the first command line argument is not a valid directory
- the second command line argument is not a valid number

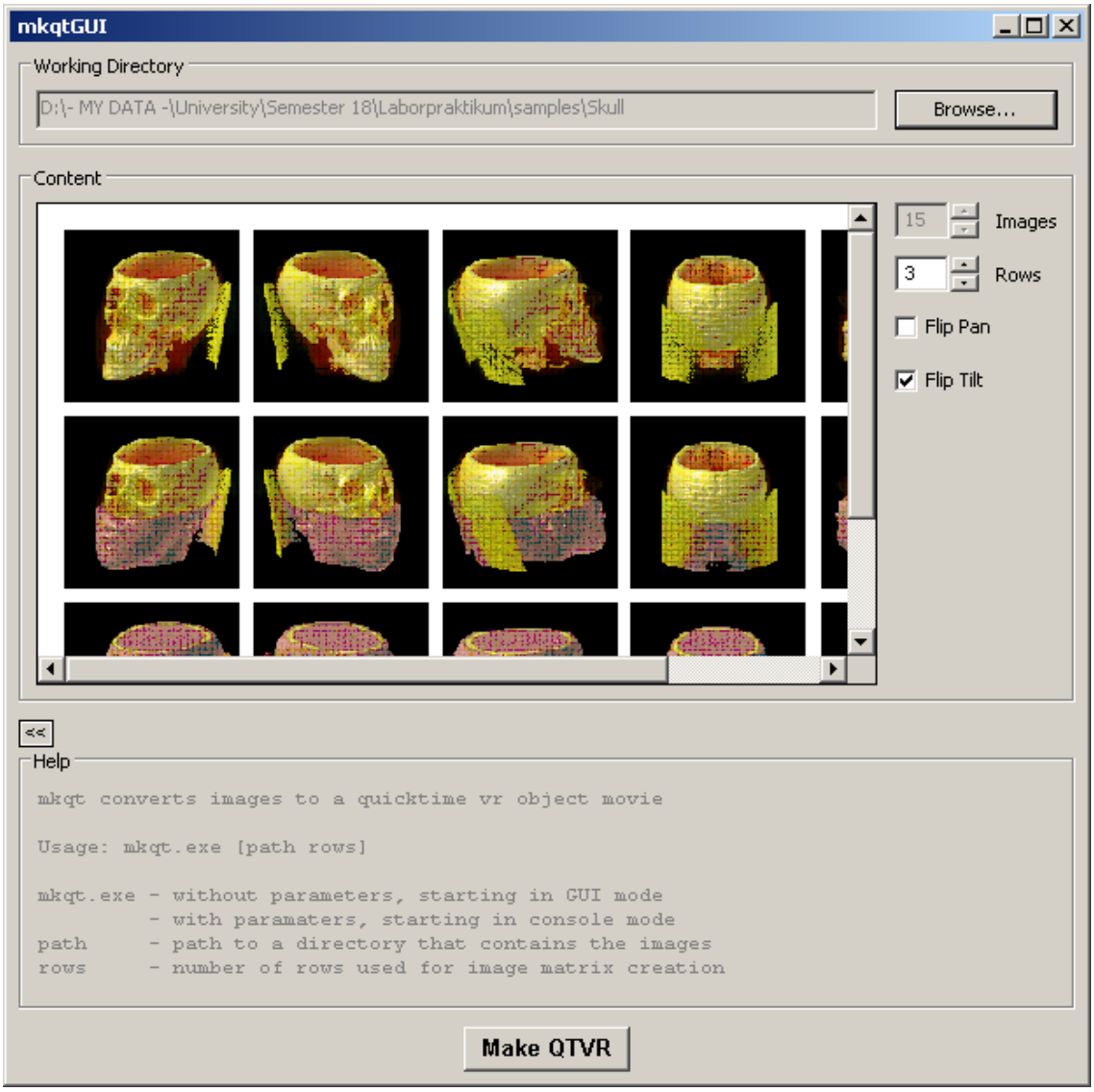


Image 03: mkqt GUI mode

When the GUI mode is started the user can select the directory which contains the images by clicking the browse button. The preview panel shows thumbnails of the images represented in a matrix that will be used when exporting the images into a qtvr movie. In the right panel are some modifiers which will change the image matrix. The preview is instantly updated when a paramete is changed.

The **Row** count defines the number of rows in the image matrix. The columns are calculated from the number of images divided by the row count. If a row number is enterd, which is not a valid divider for the number of images, a warning message appeares in the preview area. The pan and tilt checkboxes can be used to invert the image order in the matrix. **Flip Pan** will reverse the cloumns, while **Flip Tilt** will reverse the rows. After setting up the image matrix click **Make QTVR** and a "output.mov" file will be created in the same directory where the images are stored. In the left bottom corner is an additional **»** button. This button shows or hide a quick command line argument description of mkqt.