Realtime H.264 Encoding and Decoding Using FFmpeg and x264
A Quick Overview[*]

Qin Chen
Advisor: Prof. Dapeng Oliver Wu

Dept. of Electrical & Computer Engineering, University of Florida,
Gainesville, FL 32611, USA

1 Introduction

This documentation serves as a quick overview on the first development phase of our wireless video compression and transmission testbed, i.e. H.264 realtime encoding, transmission, and decoding using two open source projects FFmpeg and x264.

FFmpeg is a comprehensive multimedia encoding and decoding library that consists of numerous audio, video, and container formats. For H.264 video encoding, FFmpeg uses external open source library x264. In our testbed, FFmpeg works as an encoding and decoding front-end, while x264 is the encoding engine. At the encoder side, frames are captured by camera and then encoded into H.264 bitstream in realtime. Encoded packets are transmitted over UDP packets to the receiver. At the receiver side, packets are received, decoded, and displayed in realtime too. We have also pruned FFmpeg such that source codes of unrelated codecs and formats have been removed from the package. It will make our future development easier.

In following chapters, we will give brief introduction on FFmpeg and x264, as well as necessary external libraries in Section 2. We will also introduce how to use our software to perform realtime H.264 encoding, transmission, and decoding in Section 3. This documentation is not intended to be comprehensive. Instead, hopefully it will provide enough information to interested readers to quickly set up and try out the software, or even to do their own development.

2 Source Packages and Libraries

2.1 x264

2.1.1 Website and Download

x264 is a free library for encoding H264/AVC video streams. It is used by many other projects. Below is its official website.

```
Website:  http://www.videolan.org/developers/x264.html
Download: git clone git://git.videolan.org/x264.git
```

[*]Version 1.1, last updated on March 9, 2009. Email: eric.qin.chen@ufl.edu

2.1.2 Configure and Install

Following commands are used to build debug and release version of x264. After successful configurations, use "make" and "make install" to compile and install x264. Currently x264 should be compiled and installed before building FFmpeg. It would be nice to have only one building process for both x264 and FFmpeg. This is on our TODO list.

```
debug version:   ./configure --disable-asm --enable-debug --enable-pic
                             --enable-shared --disable-mp4-output
release version: ./configure --enable-pic --enable-shared
                             --disable-mp4-output
```

2.2 FFmpeg

2.2.1 Website and Download

FFmpeg is a complete open source solution to record, convert and stream audio and video. It includes *libavcodec*, the leading audio/video codec library used by several other projects. It also includes *libavformat*, an audio/video container mux and demux library. FFmpeg is developed under Linux, but it can compiled under most operating systems, including Windows. For detailed information and downloading, please visit following website.

```
Website:  http://ffmpeg.mplayerhq.hu/
Download: git clone git://git.mplayerhq.hu/ffmpeg/
          cd ffmpeg
          git clone git://git.mplayerhq.hu/libswscale/
```

2.2.2 Configure and Install

Following commands are used to build debug and release version of FFmpeg. Note that we must configure FFmpeg to use x264 library. After successful configurations, use "make" and "make install" to compile and install FFmpeg.

```
debug version:   ./configure --enable-gpl --enable-libx264 --enable-swscale
                             --enable-debug --disable-optimizations
                             --disable-stripping
release version: ./configure --enable-gpl --enable-libx264 --enable-swscale
```

2.3 Tools and External Libraries

2.3.1 Git

Git is an open source version control system designed to handle very large projects with speed and efficiency, but just as well suited for small personal repositories. We need Git to download FFmpeg and x264 source codes.

```
Website:  http://git.or.cz/
Download: http://www.kernel.org/pub/software/scm/git/
```

2.3.2 Subversion (SVN)

Subversion is an open source version control system. It is used to maintain current and historical versions of files such as source code, web pages, and documentation. Subversion is well-known in the open source community and is used on many open source projects and we need Subversion to down Yasm and SDL.

```
Website:  http://subversion.tigris.org/
Download: http://subversion.tigris.org/getting.html#source-release
```

2.4 Yasm

Yasm is a complete rewrite of the NASM assembler under the "new" BSD License (some portions are under other licenses, see COPYING for details). Both FFmpeg and x264 use Yasm.

```
Website:  http://www.tortall.net/projects/yasm/
Download: svn co http://www.tortall.net/svn/yasm/trunk/yasm yasm
```

2.5 Simple DirectMedia Layer (SDL)

Simple DirectMedia Layer is a cross-platform multimedia library designed to provide low level access to audio, keyboard, mouse, joystick, 3D hardware via OpenGL, and 2D video framebuffer. It is used by MPEG playback software, emulators, and many popular games and FFmpeg uses SDL for displaying decoded video contents.

```
Website:  http://www.libsdl.org/
Download: svn checkout http://svn.libsdl.org/branches/SDL-1.2
```

3 To Use FFmpeg and FFplay

3.1 Encoding

Encoding is done by binary *ffmpeg* in our trimmed FFmpeg package. It calls x264 library to encode frames captured by camera into H.264 bitstreams. We have modified *ffmpeg* such that encoded bitstreams are sent to receiver over UDP packets, or saved locally, or both. A common command line input is shown below. In this example, the input of the encoder is camera captured frames with CIF resolution. Frame rate is 10fps. The output is H.264 bitstream with bitrate at 50kbps. Intra frame interval is set to 100. The encodeded bitstream is both sent to localhost with port number 12349, and saved as a local file out.264 in /bs directory.

```
./ffmpeg -g 100 -f video4linux -b 50k -s cif -r 10 -i /dev/video0
         -vcodec libx264 -y -dest_ip 127.0.0.1 -dest_port 12349
         -f h264 ../bs/out.264
```

The command below only sends encoded bitstream over UDP packets:

```
./ffmpeg -g 100 -f video4linux -b 50k -s cif -r 10 -i /dev/video0
         -vcodec libx264 -y -dest_ip 127.0.0.1 -dest_port 12349 -f h264
```

The command below only saves encoded bitstream to a local file:

```
./ffmpeg -g 100 -f video4linux -b 50k -s cif -r 10 -i /dev/video0
         -vcodec libx264 -y -f h264 ../bs/out.264
```

In Table 1 we list some frequently used options of *ffmpeg*.

Option	Usage
-f	force input or output format
-i	specify input file name
-y	overwrite output file
-b	set bitrate (in bitss)
-r	set frame rate (Hz value)
-s	set frame size (WxH or abbreviation)
-vcodec	force video codec
-dest_ip	specify destination IP address
-dest_port	specify destination port number

Table 1: *ffmpeg* frequently used options.

Option	Usage
-h	show help
-f	force format
-s	set frame size (WxH or abbreviation)
-dest_port	specify destination port number
-fs	force full screen
-x width	force displayed width
-y height	force displayed height

Table 2: *ffplay* frequently used options.

3.2 Decoding

Decoding is done by binary *ffplay* in our trimmed FFmpeg package. It decodes H.264 bitstream and displays in realtime. We have modified *ffplay* such that it can both decodes local H.264 file and UDP packets containing H.264 bitstream, but not simultaneously.

The command below decodes H.264 bitstream in UDP packets assuming port number 12349 is used. Combined with the first *ffmpeg* command line example, realtime H.264 encoding and decoding will be performed on the same PC.

```
./ffplay -dest_port 12349 -f h264
```

The command below decodes local H.264 bitstream assuming the out.264 file is located in /bs directory.

```
./ffplay ../bs/out.264
```

In Table 2 and 3 we list some frequently used options and hot keys of *ffplay*.

Hot keys	**Usage**
q, ESC	quit
f	toggle full screen
p, SPC	pause

Table 3: *ffplay* frequently used hot keys.