

# Assignment 1

## CS414, Multimedia Systems (Instructor: Klara Nahrstedt)

Posted: January 25, 2008

Last Updated: January 30, 2008

### Disclaimer

Details in the following assignment are subject to change. Although no additional requirements will be added, it is possible that some current requirements might be relaxed. Clarifications and additional hints, which will also be posted on the course website, might also be added to this document.

### Introduction

This is the first assignment for your Internet Television (IPTV) project where you will experiment with access to audio/video devices with your group members. The goal will be to record content for your IPTV project that will be used later in your IPTV system. You should record several files since you will need at least two "TV-like" sources in later assignments. Then, you will build a player that will play your recorded content. In addition to the recorded content, you can also find content on the web, but keep in mind that the format of the found content (we are going to work with Motion JPEG videos and .au or .wav audio format to make it easier on you in later assignments for streaming protocols) since your player has to display MJPEG and play .au or .wav files. ***Audio and video recording are now optional.***

All assignments will be implemented on the Linux Dell machines (csil-core1.cs.uiuc.edu through csil-core25.cs.uiuc.edu) in the 0216 SC lab using the appropriate audio/video libraries (see the comments section for more details).

### Assignment Description

Implement a program *iptv* which takes arguments from a Linux command line. The main arguments for *iptv* functions such as **PLAY** and **RECORD** are "**function, type, duration, file**". For functions such as **REWIND** and **FAST FORWARD**, there will be additional parameters such as "**step**". The compression format for video should be Motion JPEG (MJPEG).

1. "**function**" argument will have values **PLAY**, or **RECORD**, or **REWIND**, or **FAST FORWARD**,
2. "**type**" argument will have values **audio**, or **video**,
3. "**duration**" argument will specify the duration in **seconds** how long you want to record/play/rewind/fast forward your clip,

4. "file" will be either **file.au** / **file.wav** (audio file), or **file.mjpg** (video Motion JPEG).
5. "step" parameter for the functions **REWIND** and **FAST FORWARD** means the number of data units you want to skip when doing rewind or fast forward functions, i.e., you will read every '*kth*' video frame backward or forward within the video file.

Examples below show the call of the individual arguments.

- The command line called: **iptv PLAY audio 5 meeting.wav** should retrieve from a local file **meeting.wav** the **audio** information and write (**PLAY**) it to the audio output device (use headphones to avoid disturbing other students in the lab) for **5 seconds**.
- The command line called: **iptv RECORD audio 5 meeting.wav** should read (**RECORD**) from the audio input device (Note: Be careful where the cable from microphone and headphones to the PC goes. Plug the microphone into "MIC IN" and plug the headphone into "HEADPHONE". ) for **5 seconds** and write the audio data to the local file **meeting.wav**.
- The command line called: **iptv RECORD video 5 shoe-store.mjpg** should read (**RECORD**) from the **video** board for **5 seconds**, compress each frame with motion JPEG compression algorithm and write the video frames to the local file **shoe-store.mjpg**.
- The command line called: **iptv PLAY video 5 football.mjpg** should read Motion JPEG compressed video from the local file **football.mjpg** and write the video frames to the screen. The duration of the operation is **5 seconds**.
- The command line called: **iptv REWIND video 5 football.mjpg 3** will open the video file **football.mjpg**, position the read pointer at the end of the file, and start to read and display **every 3rd** (step is 3) frame backwards towards the beginning of the file. After **5 seconds**, the rewind process should stop. It means that the reading process skips 2 frames and reads the 3rd frames, skips another 2 frames and reads the next frame, etc. (e.g., if the file has 20 frames, then the rewind process reads and displays the following frames in the specified order: 20, 17, 14, 11, 8, 5, 2).
- The command called: **iptv FF video 5 shoe-store.mjpg 6** will open the video file **shoe-store.mjpg**, position the read pointer at the beginning of the file, and start to read and display **every 6th** (step) frame forward (towards the end of the file) for **5 seconds**.

## Comments

- The REWIND and FAST FORWARD functions should be implemented only for video. This is consistent with real VCR applications. Specifically, when rewinding or forwarding a movie, the audio is left out and you see only some video frames on the screen.
- It is strongly recommended that once you record the video, you parse through the video file and find where each frame starts. Then you create an index file for this video file, which will include the sequence of pairs of information (frame number, starting file position of the frame in bytes). For example: index file of a

video file which has 5 frames would be : (1,1), (2, 4500), (3, 9005), (4, 13000), (5, 17300) (Note: Keep in mind that you work with Motion JPEG compressed video frames, hence each video frames might have a different size, although the size will not very much differ in this compression scheme.) This index file will then help you to implement the rewind and fast forward functionality, because as you skip every 'step' frame, in the index file you can find immediately the position of your frame that you need to display and you can read the frame immediately instead of searching for the beginning of each frame during the rewind/fast forward operations.

- The following websites contain source code libraries and/or examples that might be useful for Linux audio. Although not strictly required, you may choose to download and install one or more of these packages for use in the audio component of your program. You are also free to use source code from other sources as long as you understand it thoroughly and document it well with proper attributions.
  - Simple DirectMedia Layer < <http://www.libsdl.org> >
  - Sound eXchange < <http://sox.sourceforge.net/> >
  - rawrec / rawplay < <http://rawrec.sourceforge.net/> >
  - ALSA Audio API  
< [http://www.alsa-project.org/main/index.php/Tutorials\\_and\\_Presentations](http://www.alsa-project.org/main/index.php/Tutorials_and_Presentations) >
  - Java Sound Technology  
< <http://java.sun.com/javase/6/docs/technotes/guides/sound/> >
  - Java Sound Resources < <http://www.jsresources.org/index.html> >
- The following websites contain source code libraries and/or examples that might be useful for Linux video. Although not strictly required, you may choose to download and install one or more of these packages for use in the video component of your program. You are also free to use source code from other sources as long as you understand it thoroughly and document it well with proper attributions.
  - Video4Linux < <http://www.video4linux.net/> >
    - LinuxTV Video4Linux API Reference  
< [http://linuxtv.org/downloads/video4linux/API/V4L1\\_API.html](http://linuxtv.org/downloads/video4linux/API/V4L1_API.html) >
    - Note: The webcam drivers use Video4Linux version 1 (v4l), which has a very low-level API, rather than Video4Linux version 2 (v4l2)
  - FFmpeg < <http://ffmpeg.mplayerhq.hu/> >
    - Useful libavformat / libavcodec documentation  
< [http://www.inb.uni-luebeck.de/~boehme/using\\_libavcodec.html](http://www.inb.uni-luebeck.de/~boehme/using_libavcodec.html) >
    - ffmpeg tutorial < <http://www.dranger.com/ffmpeg/> >
  - Java Media Framework < <http://java.sun.com/products/java-media/jmf/> >
- Although you can use any programming language and source code library, please keep in mind that you must be able to access individual media frames in your code in order to implement rewind and fast forward for video. In future assignments, you will also need to access individual media frames for both video and audio to implement synchronization.
- If you do not implement video recording, which is now optional, then it is strongly recommended that you implement a utility to index video files as described above. This utility will help you implement fast forward and rewind.

- Please start early by reading the documentation and inspecting the example code. Also, please follow the project hints, which might include sample audio/video files for testing and example source code at a later date, at <http://www.cs.uiuc.edu/class/cs414/help/cs414Hints.html>.

## Delivery

Each group delivers:

- Source code of your iptv program in the particular group directory. The source code evaluation will be based on how well is your code documented. If you use some code you found on the net (You must understand the code you found and include in your code, not just blindly copy the code !!!) or in a local system directories, document it. It is very important that you give credit to people who developed the previous code. Your own code should include the following information at the beginning of each source code file (Note: your code will be used next Fall by another class so it is important that you give yourself credit.)
- Each major source file should include
  - File Name: Name of the File
  - Description: Short description what the file includes (general description, what kind of functions are embedded in the file).
  - Version: version of your code. You start with version 0 and as you improve the code, at some point you increase the version.
  - Programmer's Name: your name(s) who developed the code
  - Company/University Name: you put the name of the course, department and university you implemented the code for;
  - Date
- Each function in your source code file should have a header with information:
  - Function Name: Name of the Function
  - Description: Short description what the function does.
  - Arguments: Specification of each input argument parameter entering the function and its description.
  - Results: Specification of returning parameters exiting the function and their description.
  - Comments: some special system issues connected with this function
- Group representative(s) come at the scheduled time (we will have a sign-up sheet) between 5 and 7pm on Friday, February 8<sup>th</sup>, and shows a demo of the required programs in 0216 Siebel Center.

## Evaluation of the Assignment (100 Points)

- **PLAY AUDIO (20 Points)** - demonstration 15 points, correct answer to questions for this part 5 points
- **PLAY VIDEO (20 Points)** - demonstration 15 points, correct answer to questions for this part 5 points

- **REWIND VIDEO (25 Points)** - demonstrations 20 points, correct answer to questions for this part 5 points
- **FAST FORWARD VIDEO (25 Points)** - demonstration 20 points, correct answer to questions for this part 5 points
- **DOCUMENTATION (10 Points)** - Each group should write a short README file (1-2 page) in pdf format. In this documentation file you should specify your design/implementation document, short description of each functionality in terms of its implementation - don't repeat what is in this assignment write-up). Email this document to the TA [wconner@uiuc.edu](mailto:wconner@uiuc.edu), and also store it in your group directory.
- **RECORD AUDIO (OPTIONAL, 10 Bonus Points)** - demonstration 5 bonus points, correct answer to questions for this part asked during the demonstration 5 bonus points
- **RECORD VIDEO (OPTIONAL, 20 Bonus Points)** - demonstration 15 bonus points, correct answer to questions for this part 5 bonus points

The demonstration of the whole assignment for one group should take no more than 20 minutes.