An Energy-Aware Video Streaming System for Portable Computing Devices

Morihiko Tamai, Naoki Shibata[†], Keiichi Yasumoto and Minoru Ito
Graduate School of Information Science, Nara Institute of Science and Technology
8916-5, Takayama, Ikoma, Nara 630-0192, Japan
{morihi-t,yasumoto,ito}@is.naist.jp
[†]Department of Information Processing and Management, Shiga University
Hikone, Shiga 522-8522, Japan
shibata@biwako.shiga-u.ac.jp

Abstract

In this demonstration, we show an energy-aware video streaming system which allows users to play back video for the specified duration within the remaining battery amount. In the system, we execute a proxy server on an intermediate node in the network. It receives the video stream from a content server, transcodes it to the videos with appropriate quality, and forwards it to a PDA or a laptop PC. Here, suitable parameter values of the video (such as picture size, frame rate and bitrate) which enable playback for the specified duration are automatically calculated on the proxy using our battery consumption model. The system also allows users to play back video segments with different qualities based on the importance specified to each video segment.

1 Introduction

Popularization of portable computing devices (such as PDAs and smart phones) and wireless communication infrastructure (such as WLAN and wideband CDMA) has enabled watching video contents every time and everywhere. However, video streaming on portable computing devices consumes much more battery amount than other business applications, due to high processing power for video decoding, drawing and wireless communication. Thus, there are demands for controlling battery life depending on user requirements, so that the battery is not exhausted during the video playback, or no more than the specific amount of battery (e.g., 50%) is used by playing back the video.

In [1], we have proposed an energy-aware video streaming system which allows users to play back video for the specified duration within the remaining (or specified) amount of battery. In the system, we execute a proxy server on an intermediate node in the network. The proxy receives the video stream from a content server, transcodes it to the videos with lower quality, and forwards it to user terminals. Here, suitable parameter values (such as picture size, frame rate and bitrate) of each video which enable playback for the specified duration are automatically calculated on the proxy using our battery consumption model.

In this demonstration, we will present the system using PDAs and laptop PCs connected to wireless LAN. Users select one of the videos stored in a content server and input the desired playback duration and the battery amount used for video playback through the UI of a PDA (or a laptop PC). Through demo, we show how accurately our system predicts the battery amount required for playback of video. In addition, we will show the impact of our system which allows users to play back video segments with different qualities based on the importance specified to each video segment.

2 System overview

Power saving techniques

Since power consumption of decoding and drawing video depends on the parameter values of the video (such as picture size, frame rate and bitrate), power can be saved by reducing these parameter values. In the proposed system, video stream is transcoded by a transcoding proxy in the network (see Fig. 1).

In our system, power consumption of wireless communication is reduced by periodic bulk transfer: the user terminal (1) receives each fragment of the stream data at the maximum available bandwidth, (2) stores it in the local buffer and (3) turns off the WNIC while playing back the data in the buffer until the buffer becomes empty.

System behavior

The basic behavior of the proposed system is as follows. First, (1) user inputs URL of the video, desired playback duration, battery amount used for playback through the UI of the terminal. Next, (2) this information is sent from the terminal to the proxy node. Device specific constants of

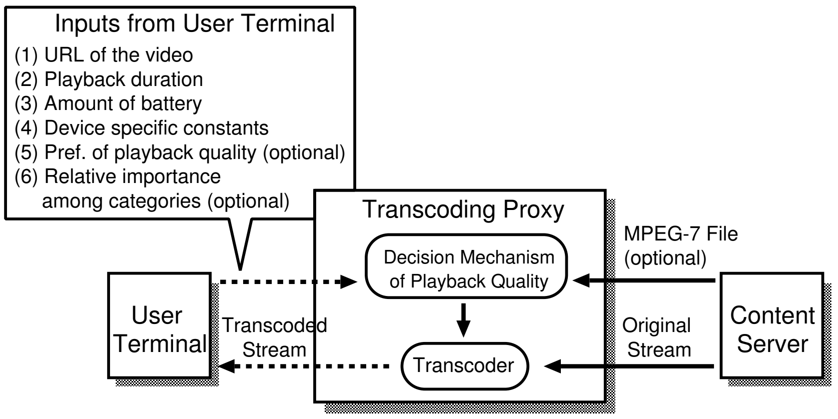
Inputs from User Terminal
(1) URL of the video
(2) Playback duration
(3) Amount of battery
(4) Device specific constants
(5) Pref. of playback quality (optional)
(6) Relative importance among categories (optional)
Transcoding Proxy
Decision Mechanism of Playback Quality
Transcoder
MPEG-7 File (optional)
Original Stream
Content Server
User Terminal
Transcoded Stream

Figure 1. System overview

each terminal (such as power consumption of playing back a certain video, etc.) are also sent at this time. Then, the proxy (3) finds the suitable quality of the video which enables playback for the specified duration within the battery amount from the quality of video sent by contents server and the device specific constants. Finally, the proxy (4) starts receiving video stream from the contents server, transcodes it to the quality calculated by the procedure described above, and transmits it to the user terminal. The video is transcoded in realtime while received from the contents server, and thus no large delay is introduced between the contents server and the user terminal.

Decision mechanism of playback quality

Decision mechanism of playback quality, shown in Fig. 1, estimates the parameter values satisfying the specified playback duration and the remaining battery amount using our battery consumption model which represents the relation among the above parameters and device specific constants of user terminal.

In our battery consumption model[1], power P consumed by the user terminal is represented by the equation $P = P_V + P_N + S$, where P_V, P_N and S denote the power consumed by playing back video, receiving video data and the other factors (such as OS, back-light, etc.), respectively. Also, P_V and P_N are represented by the equations $P_V = \alpha r f + \beta b$ and $P_N = \gamma b + \delta$, where r, f and b denote the picture size, the frame rate and the bitrate, respectively. Here, α, β, γ and δ are the device specific constants. In the proposed system, the device specific constants are assumed to be obtained beforehand using our measurement techniques proposed in [1].

Specifying importance among categories

The behavior of the system when video is played back in different quality for each scene is described next.

The proposed system assumes all scenes contained in video to be categorized beforehand. Software tools like VideoAnnEx [2] can be used to categorize scenes. VideoAnnEx can read a MPEG-1 file, identify each video segment automatically, assign a string to each segment, and output an MPEG-7 file as shown below.

```
<VideoSegment>
   <TextAnnotation>
      <FreeTextAnnotation> shoot
      </FreeTextAnnotation>
   </TextAnnotation>
   <MediaTime>
      <MediaTimePoint> T00:00:00:0F25
      </MediaTimePoint>
      <MediaIncrDuration mediaTimeUnit
         ="PT1N25F"> 78
      </MediaIncrDuration>
   </MediaTime>
</VideoSegment>
```

In the above example, string *shoot* is specified to a video segment as a category using tag `<TextAnnotation>`. Tag `<MediaTime>` describes the starting time and the duration of this segment. Here, “T00:00:00:0F25” represents the starting position of this segment is 0 hour, 0 min, 0 sec, plus 0/25 sec. Also, “PT1N25F, 78” represents the frame rate of this segment is 25fps and the duration includes 78 frames (i.e., 78 $\times$ 1/25 = 1.95 sec).

After the proxy receives URL of the video, desired playback duration, etc from user, the proxy downloads the corresponding MPEG-7 file from the contents server and sends category information to the user terminal. Then, the user specifies importance of each category through UI of the terminal. Besides importance of each category, playback characteristics (relative importance between frame rate and picture size) can also be specified. The importance of each category and playback characteristics are specified as values larger than 0, and higher value means that item is more important.

For example, suppose that a video of a soccer game consists of video segments classified into three categories $\{$*other, play, shoot*$\}$. A user may want to play back the video segments of category *shoot* at as high quality as possible, and the segments of category *play* at the medium quality, while reducing the playback quality of the segments of category *other*. The user may have the preference for playback property such that both the frame rate and the picture size are similarly important in category *other* and *shoot*, and that the picture size is more important in category *play*. In such a case, the user gives the preferences as shown in Table 1.

Table 1. Sample of user preference of playback quality

category	importance degree degree	picture size	frame rate rate
$other$	1	1	1
$play$	2	2	1
$shoot$	4	1	1

After receiving importance and playback characteristics of each category, the proxy distributes the battery amount among categories according to the proportion of the product of each category's importance degree and playback duration, and transmits transcoded stream to the terminal while receiving and transcoding video stream.

3 Implementation and Performance

We implemented the video player in the proposed system based on Berkeley MPEG Player[3]. The transcoder executed on a proxy is implemented based on MJPEG Tools[4]. The decision mechanism of playback quality is implemented in C.

Performance of transcoder

In order to test if the proxy can transcode video stream in real time, we transcoded a video stream using an ordinary PC (Intel Pentium 4 2.40GHz$\times$2, 1.0 GB RAM, Debian GNU/Linux 3.0, Kernel 2.4.27) and measured processing speed in fps. The results are shown in Fig. 2.

Fig. 2 shows that the processing speed is faster than real time in any configuration, and thus realtime transcoding is possible.

Effectiveness of power saving techniques

In order to evaluate the amount of saved battery and the precision of our battery consumption model, we played back video using periodic bulk transfer with fully charged battery, and measured actual battery life and predicted battery life. The instruments used in the experiment are shown in Table 3. In order to reduce battery consumption by spinning HDD for the laptop PC, we used noflushd[5]. Average bandwidths of each instruments are 2.3Mbps for the PDA and 1.8Mbps for the laptop PC.

Table 2. Processing speed for video transcoding

parameter value (pixel, fps, kbps)	processing speed (fps)
original video	
(640 $\times$ 480, 24, 1100)	—
transcoded video	
(560 $\times$ 420, 24, 818)	25.9
(560 $\times$ 420, 12, 788)	48.2
(320 $\times$ 240, 12, 788)	163.2
(320 $\times$ 240, 12, 294)	177.2

Table 3. Experimental environments

device type/ name/ setting	WNIC type/ name/ setting	CPU/OS
PDA/ Sharp Zaurus SL-C700/ brightness small	CF/ WN-B11/CF (I/O Data)/ low power mode off	XScale PXA250/Linux (Embedix)
laptop PC/ IBM Thinkpad s30 brightness small	PCMCIA/ GW-CF11H (Planex)/ low power mode off	PentiumIII 600MHz/Linux Kernel 2.4.27

In the case of PDA, battery life is improved from 120 min. to 340 min. by transcoding the original video with 288×216, 24fps, 327kbps to 166×124, 8fps, 109kbps. In the case of note PC, battery life is improved from 256 min. to 398 min. by transcoding the original video with 560×420, 24fps, 818kbps to 320×240, 8fps, 294kbps. Also, we confirmed that the proposed system can control playback quality adequately within the range of these battery lives.

We defined the prediction error of the battery consumption model as $|t_e - t|/t$, where t_e and t denote the predicted battery life and the actual battery life, respectively. We confirmed that the prediction error is kept below 5.3% for various playback durations and battery amounts, and this indicates that the proposed method can control battery life with practical precision to achieve the desired playback time.

Quality improvement of important categories

We investigated the improvement of video playback quality by specifying importance among categories, and compared

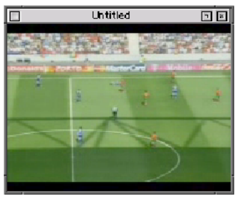

Figure 2. Playback quality of the scene $other$ **(192x144, 11.0fps, 143kbps)**

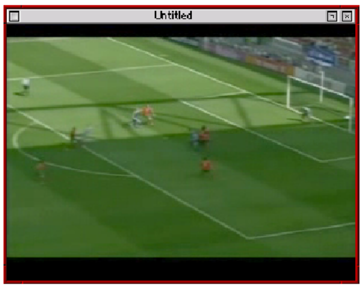

Figure 3. Playback quality of the scene $play$ **(304x216, 12.0fps, 201kbps)**

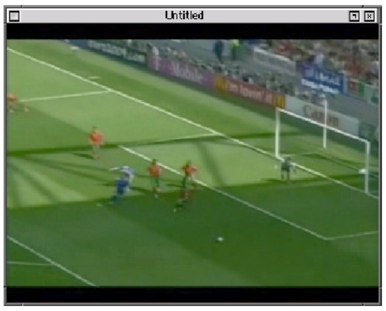

Figure 4. Playback quality of the scene $shoot$ **(320x240, 21.0fps, 365kbps)**

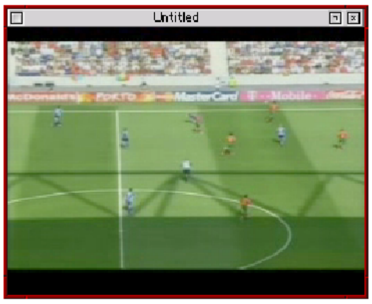

Figure 5. Playback quality w/o preference of playback quality (256x184, 17.6fps, 250kbps)

to playback at constant video quality. In this experiment, we played back a soccer video using the PDA in Table 3, and specified a preference of playback quality as shown in Table 1. Length of each category is that $other$:$play$:$shoot$ = 310 : 200 : 170 (sec), and we used 5% of fully charged battery. Playback quality of each category when importance to each category is specified is shown in Fig. 2 – 4. Playback quality when the video is played back at constant quality is shown in Fig. 5. The results show that the video playback quality for each category is well adapted according to the specified preference, and the qualities of important categories are largely improved.

Demonstration

In this demonstration, we will present our system using PDAs and laptop PCs connected to wireless LAN. Users select one of the videos (such as soccer game, etc.) stored in a server and input desired playback duration and battery amount used for video playback through the UI of a PDA (or laptop PC). In addition, users can specify relative importance among categories and preferences for the playback property to each category. Then, the system calculates suitable playback quality of each category, and transcoded video is played back on the user terminal as shown in Fig. 2 – 4.

Through demonstration, we show (1) accuracy of predicting required battery amount for playback of video and (2) the impact of our quality adaptation technique depending on importance among categories.

4 Conclusion

In this demonstration, we have shown an energy-aware video streaming system which allows users to play back video for the specified duration within the remaining battery amount. Also, we have shown the impact of our system which allows users to play back video segments with different quality based on the importance specified to each video segment.

References

[1] M. Tamai, T. Sun, K. Yasumoto, N. Shibata and M. Ito, Energy-aware Video Streaming with QoS Control for Portable Computing Device, Proc. of the 14th ACM Int'l. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2004), 2004.

[2] C. Lin, B. L. Tseng and J. R. Smith, VideoAnnEx: IBM MPEG-7 Annotation Tool, http://www.alphaworks.ibm.com/tech/videoannex

[3] The Berkeley MPEG Player. http://bmrc.berkeley.edu/research/mpeg/

[4] MJPEG Tools. http://mjpeg.sourceforge.net/

[5] NOFLUSHD - An idle-disk daemon. http://noflushd.sourceforge.net/