High Quality Lecture Recording with Minimal Bandwidth Requirements

Eicke Godehardt[1], Thomas Gabel[1]

[1]Frankfurt University of Applied Sciences
Nibelungenplatz 1
60318 Frankfurt am Main, Germany

{godehardt,tgabel}@fb2.fra-uas.de

Abstract: This paper presents a novel way to provide easy lecture recording. The actual recording is done on the hardware usually available anyhow: a laptop for the presentation and a smartphone. Voice and slides are recorded separately and displayed together in an HMTL5-based web application. This leads to minimal bandwidth requirements while still offering top notch quality of sound and graphics. Moreover the HTML5-based front-end can provide as a basis for collaborative learning and teaching.

1. Introduction

In this paper we will presents a novel easy lecture recording approach. The actual recording is done one the hardware usually available anyhow, namely a presenting laptop and a smartphone. Voice and slides are recorded separately and displayed in sync in an HMTL5-based web application. This leads to minimal bandwidth requirements without sacrificing either visual or auditive quality at all.

Of course lecture recording is not a new topic on the horizon. In the beginning everybody had unrealistic expectations on lecture recording and elearning in general. Although lecture recording is not a panacea it can really bring some benefits in elearning as well as in classroom teaching. At first it makes the students a bit more flexible in their time schedule. It can solve overlapping courses, getting the knowledge even when a student cannot attend a lecture due to sickness etc. or it can strengthen the learning effect. As our experience shows lecture recordings are especially helpful for non-native speakers of the language used in the lecture, for they can just listen to the recordings more than once. Sometimes this makes the difference for them to pass the exam or not.

The main focus of this paper is utilizing existing hardware. That means in our case a normal laptop. In addition we are going to use a smartphone for (high quality) audio recording. The second focus of this work is minimizing the required bandwidth to transfer the whole recording. How we achieved this will be explained throughout the paper.

The rest of the paper is structured as follows. We first will discuss briefly some related work of lecture recording systems. After that we will present our approach, how it differs from others and what makes it beneficial. The according implementation of our approach is presented in section 4. We conclude this paper by giving you some thought of next steps and a short summary.

2. Related Work

Today many different research work and commercial product in the area of lecture recording systems compete in the educational market. There is a huge range from simple podcast over recording of audio and slides with and without recording of the speaker up to whole digital class rooms [Mühlhäuser and Trompler, 2002]. We will discuss some of them here to give a good basis for our approach in the next section.

Most of the approaches are trying to make things easier for the speaker. And so do we. This is an important step to lower the burden of recording the lecture at first. Most of the time additional personnel is used dedicated for the recording. So many are going to focus on (semi)automatic setups. But within the whole range of possibilities several interesting questions arise. For example [Iszaidy et al., 2013] is trying to lower the necessary computation power on the server to serve many clients. But we think especially with mobile clients the actual bottleneck is not computation power but Internet bandwidth.

Other works are looking at the LMS (learning management system) integration, e.g., [Anh et al., 2010] or focusing on delivering recorded content to mobile devices [Yu and Hu, 2010]. Even though they are going for mobile handhelds, their focus is neither on minimizing of the amount of data nor the use of HTML5, but rather using a PDA to deliver learning content anywhere and at any time.

Another work is dedicated to recording of the speaker where things like face tracking and automatically following cameras are of high interest [Chou et al., 2010]. This is done in parallel to the lecture recording and users usually get both views and can emphasize one over the other—see the lecture slides centrally and the face of the speaker a little smaller or the other way around.

Most of these approaches require additional hardware for recording the actual content or tracking the speaker. For our lecture recording system no additional hardware is needed, besides a laptop or desktop computer for presentation and a smartphone for audio recording. Both is usually available today. Although it is possible to utilize an external microphone as we will explain below, this is not necessary. Smartphones in education is not new by itself. There are multiple research activities of using smartphones to get the students better involved in the lecture, like response or voting systems, but to our knowledge there is no use of smartphones for the actual lecture recording.

A last area where our solution can be compared to are screencast tools like Camtasia. The output is not that different, but as we will see later in this paper, they have not the focus on minimizing the amount of data either.

Our solution is not shown as a replacement of the approaches above. Instead it can more or less be combined and support them in providing an even better learning experience for the students of tomorrow.

3. Approach

In this section of our paper we are going to describe our approach. Figure 1 gives a general overview of how we trying to collect the different peaces of information from different sources. The square boxes represent different kinds of files or data, while the rounded boxes stand for applications, algorithms or processes.

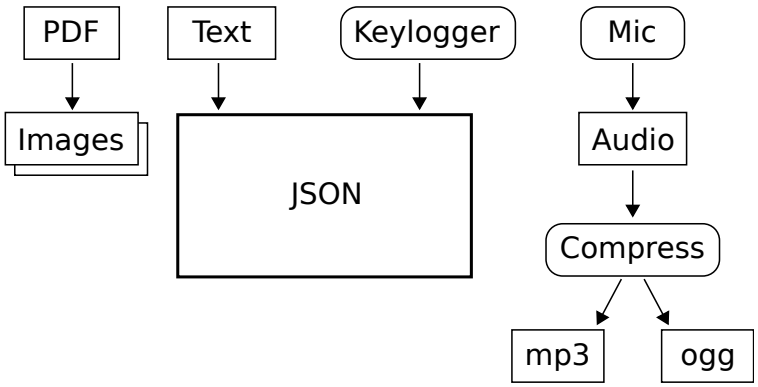


Figure 1. General Structure

Starting from the left we first going to extract all slides out of the PDF file. This way we don't have to record all activities on the presenter PC or laptop or grab the video or VGA signal. We use these images to play along the audio which is also recorded.

The actual text of the presentation is also analyzed. This is an optional step as it would enable our application to offer easy searchability within our lecture viewer. In our case we took the LaTeX beamer source files and eliminated all unnecessary LaTeX commands and splitted the text according to the individual slides or frames. Any slide may contain multiple frames, when some kind of animation is included (using the `\pause` command in case of presentations based on the beamer style). But this approach is not restricted to TeX-based presentations. All modern slide presenting applications, like Microsoft PowerPoint or LibreOffice Impress, to just name a few, use XML-based file formats. These can also be analyzed in a similar manner than what we do with our presentations.

The next two rounded boxes represent the key logging application and the audio recording on the smartphone. Keyloggers integrate into the operation system to grab all user interactions with the whole system

[Lokaiczyk et al., 2007]. We use this to extract the different user activities namely the navigation through the slides with forward and backward keys or the space bar. Whenever the user interacts with the presentation software we extract the current title of that very application. Usually it contains information like page 3 of 42. We use this information and store it together with timing information into a specific log file.

The audio is recorded in parallel on a smartphone. In our case we used a RØDE smartLav+ in addition, an affordable external lavaliere microphone developed to accomplish high quality recording using a smartphone [RØDE, 2014]. But an external microphone is not absolutely necessary. Any other head set which a user usually gets for free together with a smartphone is doing the job. The resulting audio file is than compressed and converted into mp3 as well as Ogg Vorbis. More about the issue of different audio file formats is discussed in the section about the implementation below.

In the end one central file containing all the necessary information to drive our lecture viewer is generated. It utilizes the JSON format. JSON is the Javascript Object Notation [Ecma International, 2013]. It is an easy to work with data format for machine to machine communication, and especially fit with JavaScript as the name suggests, but still is small and human readable.

In the current version of our implementation this JSON contains:

- Name of lecture
- Date of lecture
- Speaker
- Link to the audio file
- For every slide:
  - Title
  - Content
  - Link to image file
  - Timing information

It starts with the general information like name of lecture and speaker, the date and a link to the corresponding audio file. Subsequently all information about the frames are stored. For every frame that information contains the title and the actual content (for searching functionality). In addition a link to the corresponding image file is provided together with the timing information from the key logger.

4. Implementation

The implementation comprises two parts: (1) an application to digest the different input data (like PDF, corresponding text or source file, and key logger output) to produce the JSON file described above and (2) the lecture viewer which is driven by that very file. We will discuss the different parts in the following sections.

4.1. The data processing part

The first information which is to be added to the JSON file is the actual text from the different slides. We assign the text to the specific slide or frame. This serves as basis for free text search as one way for the user to navigate through the lecture.

In addition to the text source file we take the final PDF file and extract every slide as a single image. To follow our goal of minimizing the bandwidth requirements we convert each slide into the JPG format as well as the PNG format. PNG is a lossless image file format. It offers in general a much better compression ratio for pure text only slides, for great parts of the image have the very same color. On the other hand the JPG compression algorithm works best with real images and photographs.

Figure 2 gives a graphical overview on the different sizes in MByte, while table 1 show the average values of the eight different lectures converted into images, ranging from 28–75 slides per file. The outcome on this small number of lectures we used to test the conversion is not significant, but it clearly gives an indication. In some cases and constellations it saves up to 64%. In average it saved around 35% compared to the minimal standard (all slides per lecture converted either into JPG or PNG).

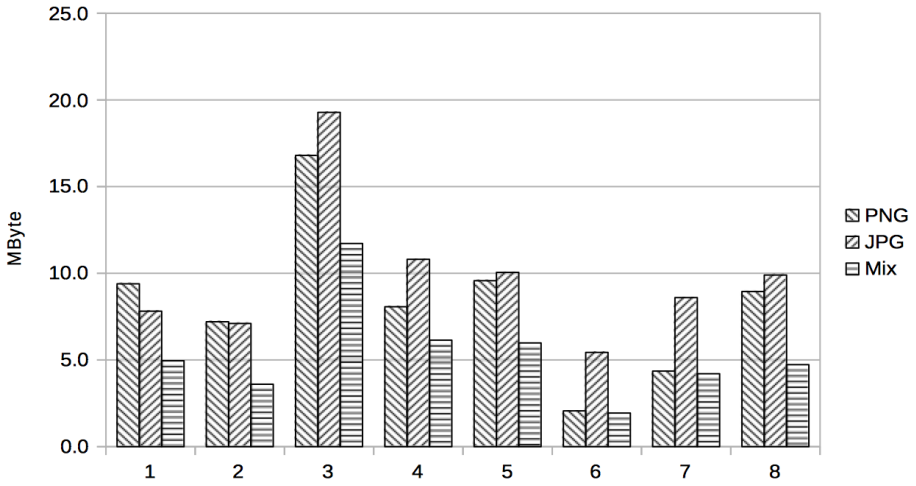


Figure 2. Comparison between PNG only, JPG only, and the best from both worlds

	Average Values
Cumulative size of JPG only	8.3 MByte
Cumulative size of PNG only	9.9 MByte
Cumulative size of smallest images	5.4 MByte

Table 1. Average compression values of converted lectures

To offer decent sound quality we follow what Richard Farrar figured out based on suggestions of the BBC (British Broadcasting Corporation) for podcasts [Farrar, 2008]. He stated that 48kbps is reasonably common for longer speech podcasts. In addition he compared that to the recommendations of the BBC for podcasts: MP3 Mono Speech (with 64 kbps, 44.1 kHz, constant bit rate). We followed this suggestion of a bit rate between 48kbps–64kbps. To support all common browsers and the supported audio codec within the `audio`-tag, we additionally convert the audio recording not just to mp3 but also into ogg [Xiph.org Foundation, 2000]. For more information regarding the different audio formats support in HTML5, we refer to [Devlin, 2011].

4.2. The lecture player

The actual lecture player or lecture viewer is based on HTML5 [W3C, 2014]. HTML5 is the current incarnation of the HTML standard. Support for HTML5 is given by almost any reasonably up-to-date browser. Two main things are making this technology especially interesting for us: (1) native support for audio and video and (2) the possibility to provide a responsive web design. The latter means an optimal support for a great range of screen sizes and resolutions—from 27" desktops monitors to small mobile devices.

We build the lecture player as an HTML5 [W3C, 2014] dynamic web page which loads the JSON file described above. As described there, this JSON file contains all information to display the audio and slides in sync.

The center builds the current slide or frame. The toolbar below provides all the normal functionality students may expect from video player, like vimeo or youtube. Another possibility which HTML5 browsers support is adjustable playback speed. This may help to listen to the lecture based on the speaking speed of the speaker and the listening capability of the student. On the right there are the different slides as thumbnails (little versions of the slides) together with the respective title. A search bar above the overview offers a content based search within the current lecture. Figure 3 shows a screen shot of our lecture player.

4.3. Comparison of our approach with normal screen recordings

In terms of the outcome our approach produces pretty similar results than usual screen recordings. So in this section we want to compare our results with the output of screen recordings regarding file size (the data is going to be transferred

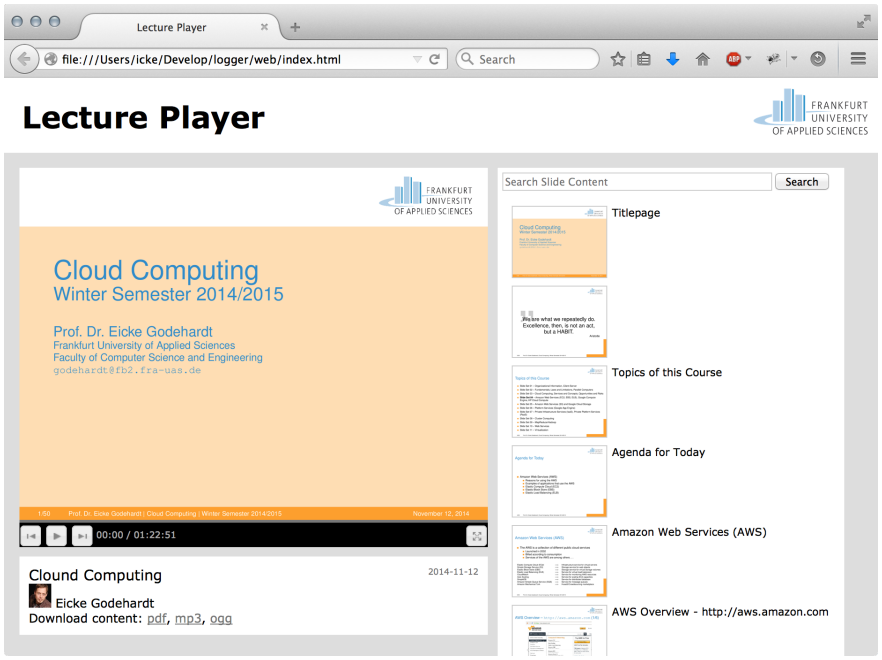


Figure 3. Screen Shot of the Lecture Viewer

to the user) and quality (visual output). In this case we used Camtasia, a well known screen recording tool. We exported the recording as an mp4 file with different sizes and quality settings.

In figure 4 we present the outputs for different sizes. As one can see, our image based approach clearly exceeds all other results in terms of visual quality. While the VGA resolution settings is still readable for most parts of the slide (only footnotes and small font is no longer readable), the slide with the resolution of 320x200 is hardly readable at all. Using 768x576 for resolution settings does not bring that much of an improvement. The small font on the footling is still not readable. Only with resolution of 1024x768 it is possible to read the footnotes, although the overall appearance is still wishy-washy. To go beyond a resolution of 1024x768 is not helpful, as most video projectors in use in universities today do not support a higher resolution anyway.

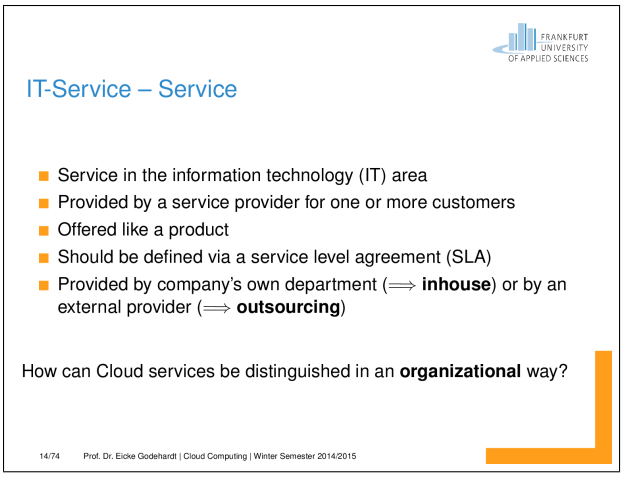
Another interesting information we get from that comparison is that changing the quality settings has no visible effect in the resulting image quality. On the other hand quality settings do effect the file size, which we want to look at now. We compared the output size based on video resolution and quality settings. Table 2 shows the corresponding output sizes to the different settings. We took the very same lecture to make the numbers comparable.

	Values for one arbitrary lecture
Our approach with Audio + Slides	54.8 MByte (40.4 MByte + 12.4 MByte)
320x200 with minimum quality	121.3 MByte
320x200 with maximum quality	270.1 MByte
640x480 with minimum quality	207.5 MByte
640x480 with maximum quality	655.0 MByte
1024x768 with minimum quality	315.1 MByte
1024x768 with maximum quality	1154.5 MByte

Table 2. Sizes compared to video out of screen recording

5. What comes next

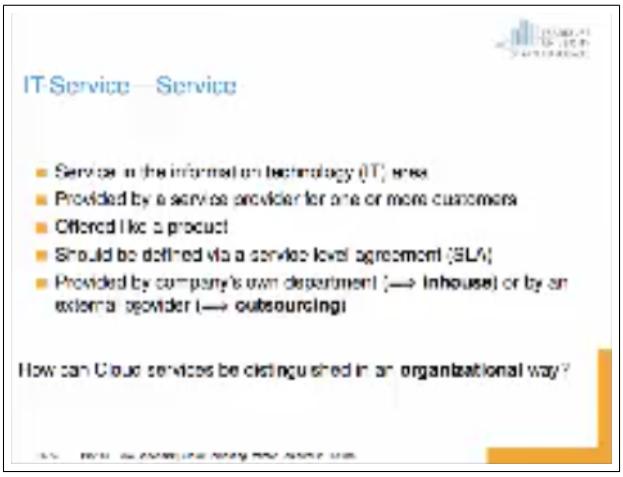
After we presented the current status of this new lecture recording system we are going to give the reader a glimpse of what is coming next in the development of our system.



Our conversion (1512x1134)

Highest non-HD resolution (1024x768)

VGA resolution (640x480)



Lowest resolution (320x200)

Figure 4. Image Quality comparison

Besides many smaller improvements there is one huge topic or area we want to address in future. It is mainly around supporting collaborative learning and teaching. We want the students become a part of the learning outcome. This will probably include feedback mechanisms, voting and several ways to feed back user response and interactions towards a more adaptive and helpful learning experience. It is all around utilizing the users to adapt the system for the better (direction of crowd sourcing). Another idea in the same direction is incorporating snapshots taken from black or white boards (maybe even from students).

Right now we only support utilizing the text on the slides itself. To offer a better search functionality we are going to use speech2text tools, to offer a search based on slide content, spoken words and maybe students comments.

We are going to offer a download possibility of the slides in PDF format and the actual audio file. This can easily extended to offer a downloadable version of a video. Using tools like ffmpeg it would be possible to recreate a video file based on the images and audio in any given resolution and quality.

6. Summary

In this paper we presented a novel lecture recording approach. Our solution achieves high visual and auditive quality while minimizing the data which needs to be transferred to the user at the same time. We think, this is the main bottleneck today when it comes to a learning experience free in time and location. We developed an HTML5-based

lecture viewer to play the slides along the audio recordings, together with multiple ways of controlling, navigation and searching.

The other focus of our work is that it does not need any additional hardware besides a laptop/desktop and a smartphone. Although an external microphone like the RØDE smartLav+ may be used for better audio quality. But even common head sets will produce reasonable audio recordings.

References

Anh, D. H., Nishantha, G., Hayashida, Y., and Davar, P. (2010). A flash-based lecture recording system and its integration with LMS. In *Advanced Communication Technology (ICACT), 2010 The 12th International Conference on*, volume 2, pages 1425–1429. IEEE.

Chou, H.-P., Wang, J.-M., Fuh, C.-S., Lin, S.-C., and Chen, S.-W. (2010). Automated lecture recording system. In *System Science and Engineering (ICSSE), 2010 International Conference on*, pages 167–172. IEEE.

Devlin, I. (2011). HTML5 Multimedia: A Review of Audio Codecs and File Formats. http://www.peachpit.com/articles/article.aspx?p=1738562, site visited: March 6, 2015.

Ecma International (2013). JavaScript Object Notation – ecma-404 the json data interchange standard. http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf, site visited: March 6, 2015.

Farrar, R. (2008). Choosing bit rates for podcasts. http://www.richardfarrar.com/choosing-bit-rates-for-podcasts, site visited: March 6, 2015.

Iszaidy, I., Ahmad, R., Kahar, N., Rahman, M., Jais, M., and Fuad, F. (2013). Embedded system for portable lecture recording system: Design and development. In *Wireless Technology and Applications (ISWTA), 2013 IEEE Symposium on*, pages 284–288. IEEE.

Lokaiczyk, R., Godehardt, E., Faatz, A., Goertz, M., Kienle, A., Wessner, M., and Ulbrich, A. (2007). Exploiting context information for identification of relevant experts in collaborative workplace-embedded e-learning environments. In *Creating New Learning Experiences on a Global Scale*, pages 217–231. Springer.

Mühlhäuser, M. and Trompler, C. (2002). Digital lecture halls keep teachers in the mood and learners in the loop. In *World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*, volume 2002, pages 714–721.

RØDE (2014). smartLav+; Lavalier Microphone for Smartphones. http://www.rodemic.com/microphones/smartlav-plus, site visited: March 6, 2015.

W3C (2014). HTML5. http://www.w3.org/TR/html5, site visited: March 6, 2015.

Xiph.org Foundation (2000). Ogg Vorbis. https://www.xiph.org/ogg, site visited: March 6, 2015.

Yu, J. and Hu, Z. (2010). Replaying lecture recordings on handheld devices. In *Networking and Digital Society (ICNDS), 2010 2nd International Conference on*, volume 2, pages 109–112. IEEE.