Abstract: We describe a system that allows recording e-learning material in the form of videos enriched by other sources of knowledge, and performing searches on the whole knowledge. Learning material can be downloaded or streamed through the web. Searches are performed through a web interface. Queries not only identify the relevant videos, but also their exact timing, so that users are able to play the appropriate portion of the movie and retrieve the associated material.

Keywords: Multimedia, Information Search and Retrieval, Multimodal Interfaces

1. Introduction

E-learning is being used in many different and even contrasting ways. On the one hand, some people see it as an opportunity to rethink teaching, generating more direct and effective student involvement. On the other hand, others consider it as a convenient way to deliver traditional teaching in a non-traditional setting (e.g. at a distance). Web conferencing offers two flavors of this second view: webcasts and webinars. Webcasts are nowadays rather common as a way to unidirectionally diffuse seminars or events, allowing to break space constraints and to remotely assist to a lecture. Remote users can (passively) participate to the event through the Internet. A variation of webcast, for which the new term “webinar” (a portmanteau that fuses “web” and “seminar”) has been coined, extends webcasts by allowing a (typically asymmetric) bidirectional interaction. Often is such case the direct channel (the one going from the speaker to the audience) is an audio stream (or less frequently a video) combined with some support learning material (e.g. slides), while the inverse channel carrying feedback and questions is typically provided through a simple chat. Webinars and webcast are often also recorded so as to allow an asynchronous exploitation (of course in such case the user is deprived of the possibility of using the inverse channel). When an event is recorded not only the temporal constraint is broken, but the user can sometime also navigate the content by skipping less interesting parts or searching some especially relevant piece of information. In general though there is not much support for navigation and even less for search.

Our work started a few years ago (Ronchetti 1983a, 1983b) by exploring the application of webcasts and webinars as a form of support for traditional lectures in an academic setting. We aimed mostly at helping people who for various reasons could not be in class, by providing a surrogate through videos synchronized with presentation slides. Quickly we discovered that most students (who attended the lectures) used the system as an integration to check the correctness of their notes and to review some small (and probably critical or poorly understood) portions of the lectures. We therefore realized that tools that facilitate the search and retrieval of portion of
videos were an important missing element, and focused our research on such aspect. The present paper describes our first prototype and results. In the next section we describe the background and our initiative for recording and making available a set of academic lectures. Later we introduce our search architecture, and finally we report some data.

2. Digital videos as a learning resource

Video streaming over the Internet is a relatively new technique. In fact, although it is possible to use some form of streaming video even at relatively small bandwidth, such technology became popular when a broader band supported last mile connections of more than one gigabit per second. Nowadays many users upload their videos to popular sites like Google Video or YouTube. Much has still to be done to solve the problems of organization and search (and especially semantic search) of these resources. The use of streamed digital videos as a teaching and learning resource became in recent years an attractive option for many educators. They can expand the range of learning resources available to students by moving away from static text-and-graphic resources towards a video-rich learning environment. This environment can offer images, interactivity and integration with other resources. It can integrate, inter alia, still and moving images, live or recorded lectures, locally produced video, web resources and synchronous and asynchronous communication tools. Thus, streamed video allows remote access to lectures and, when integrated into a multimedia package, creates a rich, accessible, interactive and controllable teaching resource.

Streamed video is already widely used in a few universities. For example, the College of Engineering of the University of Cincinnati (web 2006) offers several complete courses via video streaming. It sees video streaming as successfully replacing the classroom and as drawing the wider learning community and the university closer together. At the University of Western Australia (Fardon et al. 2005) video streaming has given over 6000 students access to recordings of over 1800 lectures. 50000 hits have been recorded to date - 60% of which came from off campus. The University of Sydney (Wozniak et al., 2005) is also using streamed video to deliver lectures both synchronously and asynchronously. Several similar cases are available: however, in these examples streamed video is mostly being used for transmitting unenhanced recordings of live lectures.

Since 1993 we have been making available recorded video lectures at the Science Faculty, Università di Trento. We started by using a software called e-Presence that had been developed at KMDI, University of Toronto (Baecker et al. 2003), and later developed our own software called LODE to better respond to some special requirements we had (Dolzani et al. 2005).

E-presence allowed us to webcast audio, video and synchronized slides, and also pioneered the inverse channel that allowed some interactivity through chats. However, our main interest was not to allow students to attend lecture from outside the classroom in real time - we saw only a limited value in this, and in fact only 1% of the students used the system this way. The most important added value we were seeking was in breaking the temporal constraint, allowing students who are not free at the lecture time to attend the regular lectures. We were aiming at supporting the worker students who typically cannot benefit of the lectures because during the day they work, and typically study in the evening and during weekends. Such students typically have no other option but to use books, notes taken by regular students and additional reading material provided by the teacher. Although the percentage of such students in not very high, being approximately 10%, we
thought that it was worth attempting to improve the quality of their study, and that in such way we could also attract more students.

The idea of providing video lectures for distant students, or for students that have time constraints is not new. In most implementations however this is done by recording ad-hoc lectures. In our opinion this has several important drawbacks: in first place it is very costly. Special lectures are prepared by teachers, and they are recorded in a TV-like production environment. The resulting effect is often a talking head that provides a very flat and boring synthetic lecture.

Since the very beginning we had a clear view that what we needed was an inexpensive way to produce lively video lectures. The solution was easy: recording real lectures provides a learning experience that is much better than the one obtained by artificial lectures. In a classroom the teacher has a constant feedback from the present students: s/he can choose to modify the lecture’s pace, to tell a joke to relax the atmosphere, to repeat a concept with different words because s/he realizes that (part of) the students are having difficulties in understanding. A video recorded from a real lecture is therefore pedagogically much better than one constructed in an artificial environment. The lecture takes place anyhow, so there is no additional cost related to the teacher. The problem here is that not everybody is ready to be recorded during a lecture: making the video available is a sort of publication, and many people would like to have control over that process. Spoken language is less formal than written language, and it is inevitable that a less rigorous wording be used. Even mistakes happen, but while in class they just “pass away”, having them recorded is certainly a worry. Synthetic lectures can be adjusted, but correcting a real lecture is much more difficult. Moreover recording makes the class walls transparent: colleagues might see the lecture and judge it. To accept to be recorded during a lecture takes a degree of self-confidence, and the presence of a camera might take away some of the sought spontaneity. To alleviate the matter, we offered the possibility of destroying the video of a lecture without asking question if the teacher was not happy with the result. No one ever requested it. Also, all the teacher that participated in the initiative declared that they were not bothered by the presence of the camera.

The development of our own LODE system was guided by some basic requirements: we wanted to achieve minimization of acquisition and running costs, easy transportability of the acquisition system, quality of the videos, support of playback in generic OS platforms (Linux – Apple Macintosh – Windows). As far as costs were concerned, we based the whole system on free software, we made the post-processing of the videos fully automatic, and we created a simple interface for the video operator, so that unskilled (and inexpensive!) new operators can be trained quickly. Typically one of the students attending the lectures is in charge of recording. Therefore the system must present a low cognitive overhead so that the student can actually follow the lecture while performing the service. We also minimized the requirements for the hardware, so as to reduce cost and size of the station and to have a short setup time (the acquisition system simply consists of a laptop with Firewire connection plus a digital camera and a radio microphone, for a total cost of less than 2000 Euro). As far as the video quality goes, we based the LODE system on MPEG-4, as it offers the possibility of coding video at a good resolution while maintaining a relatively low bandwidth occupation. This allows us to have good quality video with a resolution of 550x440 pixels, which is large enough to support non-PowerPoint based events like blackboard writing. The client does not need to buy anything, since the Apple QuickTime plug-in for browsers is free for Apple Mac OS-X and for MS Windows. At present a porting
of QuickTime on Linux is not available: however in the Linux arena other
MPEG-4 players are available and the same is true for UNIX, so that our
system can have clients on all major OS platforms. The video-only portion of
the lecture can be viewed also on an inexpensive mp4 player (Figure 1), so
that users can assist at a lecture even without a laptop, and for instance
while commuting. We are presently attempting ports on the last generation of
cellular phones.

![Figure 1 – LODE’s video running on a small mp4 player](image)

Among the other features, we mention the ability to suspend and resume the
recording of a lecture (e.g. to allow students to make an exercise in class),
and the option to cut a lecture into pieces and recompose them according to
semantics rather than to temporal sequence (this helps reusing lectures over
the next year, or cutting out unwanted sections). Finally, it is possible to
attach arbitrary documents to any time of the video.
The Lode output is an archive for each lecture with a standardized directory
structure. The directories contain the video recording in MPEG 4 format, the
images of each slide as jpegs (or other formats usable in a browser), an HTML
file and a JavaScript file for enabling dynamic interaction between video and
slide presentation. Inside the HTML file we keep the time information that
allows the slide (or other resource) synchronization with the speaker
presentation.
The user views the lectures through a standard Java and JavaScript enabled
web browser (Figure 2). S/he can navigate the lecture by using either a time-
bar that has marks that show relevant events (like the change of slides, or
the presence of attached material) or through an automatically generated set
of slides’ titles. Although this resembles other the product of other
existing systems (like Microsoft Producer) we outline that here no human-
supervised post-processing is necessary.
The user can (at any time) switch among three different resolutions: 550x440,
360x288, or 180x144 pixels and correspondingly allocate less or more space to
the slides (if present). The small resolution allows allocating more space to
the accompanying image (typically a PowerPoint slide). The largest resolution
allows reading what the teacher writes on the blackboard, while strongly
reducing the size of the accompanying image, which in any case can be
enlarged to full-size in a pop-up window.
Using the system over a few years allowed us to construct a repository of video lecture that presently contains more than 500 hours of video lectures and seminars with associated video-synchronized PowerPoint slides. These lectures can be accessed via web, or obtained on DVDs: one single DVD can contain all the recordings of an extended event, such as a Summer School or a 50 hours course.

At present our multimedia database comprises 17 series for a total of approx. 260 Events corresponding to a total of approximately 500 hours of recordings and 50 speakers (30 in English and 20 in Italian). Most of the topics (88% of the total time) are in Computer Science, but there are some on different areas (Meteorology, Sociology, Engineering). 64% of the speakers and 38% of the hours refer to events in English, 36% of the speakers and 62% of the hours to Italian.

Details are the following:
3 International summer schools on Computer Science (Semantic web, Web Engineering) for a total of 80 hours and 30 speakers. In English;
2 Workshops (“Human Language Technology and “Noise on the workplace”, 15 hours, the first in English the second in Italian;
2 Schools on Meteorology, 60 hours, 15 speakers, partly in Italian and partly in English;
8 Bachelor courses in Computer Science, 250 hours, 7 speakers, in Italian
2 Master courses in Computer Science (Web architectures and Machine Learning), 80 hours, 2 speakers, in English
1 Master course on Sociology of Tourism, 20 hours, 1 speaker, in Italian
As we mentioned, the initiative was originally targeting mainly worker students, but soon we realized that it was a great value also for regular students. They reported using the system to review mainy portions of lectures while preparing for the exams. The system was used by a very large percentage of students, and by virtually all the students who took the exam at one of the later calls. It is worth mentioning that in the Italian academic system students can take the exam right after the course, or in one of the following calls - they have up to five or six calls during the academic year- and if they fail they can take the exam again at later calls. The best students usually pass the exam right at the first call, but a not too small percentage of pupils drags exams even over the following years, so that it is not too uncommon that for a three academic years course someone needs five solar years. We found out that the availability of the videos helps most the students who cannot make it at the first call - so we’re actually helping the most exactly the ones that need more help.

3. Performing multimodal search on videos

Finding out that students are mostly interested in reviewing portions of lectures prompted us to investigate which kind of support can we offer to best support their need. As we mentioned, the recorded lectures are navigable through a time bar that has some semantic marking on it (at minimum an indication of when slide transitions occurred - and when the user moves the mouse over the marking bars, a popup shows the title of the corresponding slide). Also, all slide titles are automatically extracted (for Microsoft PowerPoint presentations) and presented on the user interface, so that navigation is also possible by clicking on such titles.

Finding when the teacher mentioned some particular topic might be more difficult if not almost impossible for the user. An example of such scenario would be trying to find out when, in a 50 hours course, the teacher described how the final project has to be performed. Chances are that there are no dedicated slides, and that the issue was introduced in the middle of some lecture, and maybe in one of the following lectures discussed again to respond to some students’ questions.

The resources we have are the slides, possibly some additional material and the video, which is actually composed by the audio and the video tracks. This is a perfect scenario for multimodal search.

In video indexing domain Snoek and Worring (Snoek 2005) proposed to define multimodality as “the capacity of an author of the video document to express a predefined semantic idea, by combining a layout with a specific content, using at least two information channels”. The channels or modalities of a video document are described in their work as:

- Visual modality, the video image content, everything that can be seen in the video;
- Auditory modality, the speech, music and environmental sound; everything that can be heard in the video;
- Textual modality defined as the resources that describe the content of the video.

In a lecture context the visual modality is not of great help. The scene almost never changes, the transitions being related to a change of modality (from slide to blackboard and back) or to the change of slide. We already have markup of slide changes that is acquired during the lecture acquisition, so not much else is left. Teacher gesture and facial prosody might give some
help, for instance in determining what the teacher believes being the most important passages, but we decided not to attack such problem during a first stage. We already used, at least in part, textual modality by extracting the slides’ titles. We decided to extend this by indexing the full content of the slides, so as to be able to perform textual searches. The auditory modality was for us the most promising. Analysis of the speech can be done at signal level, finding for instance patterns that may be significant for instance for capturing emotional features, or at a higher level by identifying phonems and recognizing words. We focused on this last aspect, and applied automatic speech recognition (ASR) to the movies soundtracks. In particular, we were interested in the STT translation (speech to text). The output of the STT process is a transcript of the speech with a temporal markup, i.e. every transcribed word is marked with the time at which it occurred. STT is not perfect, and in fact there have been studies investigating the readability of STT results in various contexts (see e.g. Jones 2004). So the challenge was to investigate how good the transcript were, and how effective are they in the task of information retrieval.

4. Speech recognition experiments

The speech recognition tools we have evaluated are a speaker independent transcription system built by a research group (in the following called R-ASR) and a commercial dictation system (in the following CDS). The domain of the content to be transcribed is Computer Science, and in particular we worked on lectures that introduce the Java programming language to first year students. We used the tools to get the transcripts from these lectures, that were recorded at the University of Trento in the Spring semester of 2006. The spoken language was Italian. We used an open source software called Transcriber\(^1\) for segmenting, labeling and transcribing the speech flow that was recognized by the ASR. Transcriber produces a transcription file, and can open and edit it. A transcription file is an XML file that contains sentences aligned with a temporal tag. In order to have a comparison benchmark, we generated a manual transcription of the lectures so as to have a “correct” interpretation that could be used for measuring the performance of the ASR tools. The comparison between the transcript generated by the ASR tool and the manual has been done using the Speech Recognition Scoring Package (SCORE) Version 3.6.2 tool\(^2\). SCORE takes two transcripts in input and can generate various types of reports. The R-ASR tool was not trained on the specific domain. It had been previously trained to be used in the Italian parliament. The commercial tool (CDS) was trained with its default training set. The first part of the experiment was carried on using the R-ASR on just one hundred sentences randomly picked from one lecture. A manual transcription has been done on the selected sentences so as to obtain the correct reference set. The reference set and the corresponding phrases as recognized by the ASR tool were given to the SCORE tool, which generated a report that contains information extracted from the comparison of the two transcriptions and a file containing the system alignment structure. The system alignment structure tells for each compared sentence if a word is inserted (I), missed (D) or substituted (S) with another one with respect to the reference set.

---

\(^2\) http://www.nist.gov/speech/tools/index.htm
The report file generated by SCORE gives information about the performance of the ASR tool. The analysis is focused on sentences and words. We obtained that 60% of the words were correctly recognized with a word accuracy of 56.7% (corrected – inserted).

We’ve found that the main problem lies with technical terms that were never recognized. Without training, or without a domain vocabulary, no ASR would ever recognize the word “Java” in an Italian phrase.

In the second part of the experiment we proceeded with a vocabulary analysis and we add technical terms to R-ASR vocabulary. The aim of this part was to verify if a domain dictionary could increase the performance of the tool. In the following we describe in details the step we’ve taken in running the experiment.

The first step was the word extraction from the manually transcribed one hundred sentences and from the R-ASR vocabulary. R-ASR’s vocabulary contained 62,879 words (A) while the manual transcription had 702 terms (B).

The next step was to match the enhanced R-ASR dictionary against the words extracted from the manual transcription. We found that after this operation the R-ASR dictionary was lacking 57 of the words from the manual transcriptions. 35 out of 57 of these words were technical terms about computer programming in general or Java. The remaining words were: 4 terms with prefix ri (re in English) (“ricompilarlo, riguardiamo, riparla, riusare), 3 company or product names (“Fineco, BeOS, Macintosh”), 3 Spanish words, 3 reflective verbs, 1 English word (“embedded”), 1 diminutive (“bellina”), 6 verbs and the mathematical word “factorial”.

To analyze the domain vocabulary we used three freely downloadable electronic books on the same domain of the recorded lectures. The three books’ vocabularies had respectively 5,516, 9,004 and 1,995 words. Their union counted 11,443 words. We added these terms to the R-ASR dictionary to try to cover the technical words that were missing in the original R-ASR dictionary. The final step was to run the speech recognition tool again. As expected, the enhancement of the vocabulary leaded to an increase in the number of correct recognized words. As a result of the vocabulary extension, the percentage of un-recognized words decreased from 8% to 3%.

Another experiment was to run both speech recognition tools (R-ASR and CDS) on an entire lecture. We followed the same procedure we’d used in the first experiment. We have compared the transcripts generated with the speech recognition tools against a reference set produced by a manual transcription. The results show a higher percentage of errors. We have carried out a vocabulary analysis and we’ve discovered a big lack of terms in the dictionary. The lecture we’ve used in the second experiment was an advanced lecture; this means the number of technical terms was much bigger than in the lecture we used in the first experiment: being an introductory lecture, it contained more general terms.

The dictionary is therefore a fundamental factor for the improving of the ASR tool performance. We found a 36% of correct recognized sentences using R-ASR tool.

CDS tool performed better in the experimental domain. The percentage of correct sentences is 76.6%. One should note however that CDS was explicitly trained on the speaker, while R-ASR was a speaker independent tool.

One should however note that for our intermediate goal (i.e. indexing the content of a lecture for performing google-like word searches) word accuracy is far more important than correct sentences (and of course the percentage of correct words is higher than the percentage of correct phrases). For a longer term goal (e.g. semantic matching) sentence accuracy would be more relevant.
5. Needle. System architecture and prototype

Up to now we described the video acquisition and playback system, our library of video lectures synchronized with external documents, and the (imperfect) transcripts of their audio tracks. We also declared our intention to perform multimodal search. Let’s therefore come to the description of the system that leverages our assets and makes it possible to perform google-like searches on the videos: Needle.

Needle is a software we developed for searching among the multimedia learning material collected by LODE. The acronym stands for “Next gEnEration Digital Library”. It aims at allowing the user to perform a textual search that finds and retrieves relevant portions of the videos contained in a repository. The application allows users to formulate text query against our multimedia knowledge base and displays the results. The query is run against all available data sources: the transcripts, the content of the slides, the content of any additional material attached to the lecture. Since all these resources have temporal markings, any hit allows finding a video resource and a time that allows positioning it. In such way, searching for the word “exam” would find all the occurrences of the word in the lectures, and to quickly retrieve the relevant portions of the videos. Searches can be refined and made more effective by using standard tools like Boolean and proximity operators. Of course errors in the STT affect the results, but we found that even poorly optimized ASRs produce a very effective starting point.

Needle was designed as a web tool, assuming that the user would run queries against a streaming server that delivers the relevant portions of the videos. It is composed by a web user interface and a server side part. The user platform requirements are minimal, being the playback based on the same design as LODE: only a standard, Java and JavaScript-enabled browser plus the Apple QuickTime plug-in are required. The server side part enables the searching and ranking on learning materials. Needle was developed in Java and
it has been designed to ensure portability and usability on different platforms.

The learning resources come from different sources. The video lectures' recordings originate from the LODE system: however the link to LODE is not structural: any mp4 video could be processed through STT with minimal process variations. Of course the advantage of the LODE source comes from the multimodality given by the additional documents.

The reference for each lecture to the LODE static directory structure is stored in our knowledgebase. We extract the audio track from the movie, and we save the link to it in the media storage. Using ASR tools we extract the transcript from the audio file. The transcript, the presentation slides and other learning materials are saved in the knowledgebase as well.

The middleware layer is composed by modules for accessing the data sources, and for searching on them (Figure 3). In our prototype the search engine module performs a text search on the lectures' audio transcriptions and on the slides content. An upper level module is in charge of merging the results that have been found in the different media (e.g. in the slides and audio files) and also provides some basic ranking functionality. We layered the application upon the full text search engine Jakarta Lucene\(^3\). The multimedia engine module acts as an interface between the web client and the data sources.

![Needle interface](http://lucene.apache.org)
The user interface is a web page with an input field for entering the search string. The string can be a single word or multiple words. Logic operators, wildcards and proximity operators are allowed. The user can also restrict the search to one class of data sources (e.g. only from video transcripts). The result page displays the results arranged by course and lecture (Figure 4). For each lecture, results are shown inside a table that allows the user to easily understand which results belong to each lecture. On the left most part of the table an icon shows in which media the result has been found. In our prototype the result can be found in the slides or in the video. Right after the icon there is a text that contains the searching word inside the part of the context where it has been found (for video hits, the textual description would be a part of the transcript file, in the case of slide hits it would be part of the text present in the slide). The rationale of this choice is to give the user a first indication of the context, even without opening a video (which is a slower operation than reading a phrase), and therefore to help finding he most relevant hits. All the searched word are highlighted inside the textual description. In case of multiple words, each word is highlighted with a different color.

After the textual description four modes for accessing the multimedia results are displayed. The first link displays the video segment of the lecture in MPEG 4 format. For avoiding network problems we display just the segment of the video where the result has been found: the user has anyway the option of viewing the entire video or the subsequent segment. The second link points to the audio file of the lecture in MPEG 3 format. Both audio and video files are positioned in correspondence of the beginning of the sentence containing the found word, and are displayed with a navigation bar. If the hit regards a slide, video and audio are positioned to the first time in which the speaker showed the slide.

The third link is for displaying the slide that is temporally correlated with the video and it may contain the search hit. Also the slides are displayed with a navigation bar that allows going back to the previous or forward to the next slide.

The forth media mode is a link to the LODE user interface. Also in this case the system is synchronized with the time corresponding to the hit.

6. Conclusions.

The Needle system we have presented in this paper is our first attempt to answer to the need of accessing an un-structured multimedia knowledge base of learning materials. Our starting point is a database of more than 500 hours of recorded lectures, enhanced with presentation slides and additional materials, collected through our LODE system. Needle provides an interface for searching inside the video lectures and it presents the results in a user friendly way. Needle can be considered as an e-Learning tool for supporting students in their learning task. In particular, it allows them to easily find material in the large body of a course. We think Needle could lead to an improvement of study efficiency since the searches on different media can integrate the information that could be presented in a single modality. Needle allows searching in the spoken flow: Even though the transcript is only as good as the ASR makes it, its quality is sufficient to perform searches that yield reasonable results.

Our approach to video indexing and search is multimodal. We focus not only on the video itself, but also on the related material that comes with the video-lecture, such as presentation slides and reference documentation. Information
is not carried by just one single media but can be spread in different media. Needle finds the in different media types. The system displays the media where the results have been found and it also shows the correlated learning materials that were presented at the same time during the frontal lecture. The tool could search inside a potential collection of heterogeneous materials (desktop activities recording, PowerPoint presentations, interactive whiteboard tracks, forums etc if marked with temporal reference to the lecture). We believe that enabling searching on different multimedia types would lead to a more effective learning: the aim of the project is to increase flexibility and efficiency in the study. A first evaluation that we carried on (Fogarolli and Ronchetti, 2007) seems to support our expectations.

We have faced a number of challenges in the successful development and deployment of our multimedia search engine. The performance of a speech recognition tool is an important factor for the improving of the information retrieval, but even a ASR that would be inadequate for extracting reading material from the movie soundtrack can be rather effective for the indexing task, since a reasonably good word accuracy is enough (even if the percentage of correct sentences is low).

Data organization and retrieval were successfully approached by using free-text search engines. Since all our resources have temporal marking, from the found entities we can retrieve both the correctly positioned original video and the related material.

Low-level feature recognition (both on audio and video) could bring some additional benefit, but we think that focusing on TTS was the best and simplest choice. Future work might include exploring these possibilities.

This work can also be relevant in the digital libraries context and as an advanced knowledge management tool.

7. Acknowledgments

We thank Alessandro Bertacco for the manual transcriptions and for his work in the initial analysis of the data.

References.


