A Multimedia Retrieval System Using Speech Input

Andrei Popescu-Belis
Idiap Research Institute
Rue Marconi 19, BP 592
1920 Martigny, Switzerland
andrei.popescu-belis@idiap.ch

Peter Poller
DFKI GmbH
Stuhlsatzenhausweg 3
66123 Saarbruecken, Germany
peter.poller@dfki.de

Jonathan Kilgour*
HCRC, Univ. of Edinburgh
10 Crichton Street
Edinburgh EH89AB, Scotland
jonathan@inf.ed.ac.uk

ABSTRACT

The AMIDA Automatic Content Linking Device (ACLD) monitors a conversation using automatic speech recognition (ASR), and uses the detected words to retrieve documents that are of potential use to the participants in the conversation. The document set that is available includes project related documents such as reports, memos or emails, as well as snippets of past meetings that were transcribed using offline ASR. In addition, results of Web searches are also displayed. Several visualisation interfaces are available.

Categories and Subject Descriptors: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval.

General Terms: Design, Human factors.

1. INTRODUCTION

The availability of recording and storage devices facilitates the acquisition of multimodal databases for use in personal or corporate contexts. The challenge now is to provide meaningful access to such collections, which otherwise remain largely unused. With the goal of lowering the cost of consulting such data in terms of a user's cognitive load and access time, we introduce a *query-free* or *just-in-time* retrieval system [2, 3], the AMIDA Automatic Content Linking Device (ACLD), which gives access to chunks of multimedia recordings based on the information context that is sensed using automatic speech recognition (ASR). The ACLD can be used during a group discussion to provide access to past information, at a minimal attentional cost for the participants.

2. SCENARIOS OF USE

Participants in a meeting often mention previous discussions or documents, but accessing them requires more time than participants can afford to spend during the discussion. Our system provides an efficient solution for searching past meetings and documents, thanks to which participants only need to decide if they want to consult (and possibly introduce in the current discussion) the meeting fragments or documents retrieved automatically for them. In the intended setting, our system can either be used privately by every participant to a meeting on their individual computers, or its results could be shown to the entire group on a separate projection screen.

For demonstration when no meeting actually takes place, the system can be run using a pre-recorded meeting, by playing it back and feeding its signals into the system as if they were live. For this purpose, the AMIDA Meeting Corpus helps us considerably. We use a group's last meeting from each series (a meeting labeled with 'd', e.g. 'ES2008d') to simulate a live meeting, treating segments from the three previous meetings (labeled 'a', 'b' and 'c') and associated documents as the group's multimedia history to which our system provides just-in-time access. This makes it possible, also, to replicate the same behaviour over and over again, a crucial property during software development and debugging.

When demonstrating with a pre-recorded meeting, the ASR can either be simulated from offline-processed data, or the ASR can be run in real-time. However the difference between the two will not be immediately visible to the audience – unless the ASR output quality (lower for online ASR) is carefully examined. To demonstrate the live ASR feature, a version of the system can be run using the speech from the presenter of the demo, through a headset microphone, on condition that a separate computer is available to run the ASR system. In this case, the retrieved results depend on the speech content and noise conditions.

The data used for development is the AMI Meeting Corpus [1], which consists of 100 hours of recordings, in English, from an instrumented meeting room. Groups of four people carry out a role-playing exercise, over four meetings noted 'a' through 'd', in which they pretend to be a design team specifying a new kind of remote control. Final design decisions are made in the fourth meeting of each series. A number of project documents and fragments of previous meetings are available and may be relevant to the discussions in the current meeting: reports, emails, and presentations given during the first three meetings, plus multimedia segments (video, audio, transcript) derived from the first three meetings by dividing them into one-minute chunks.

3. COMPONENTS AND INTERFACES

In a nutshell, the ACLD performs searches at regular intervals over a database of "documents" derived from the AMI Meeting Corpus, with a search criterion that is constructed based on the words that are recognized automatically from an ongoing discussion. The results are displayed by user in-

*Additional authors appear at the end of the paper.

ICMI-MLMI'09, November 2–4, 2009, Cambridge, MA, USA.
ACM 978-1-60558-772-1/09/11.

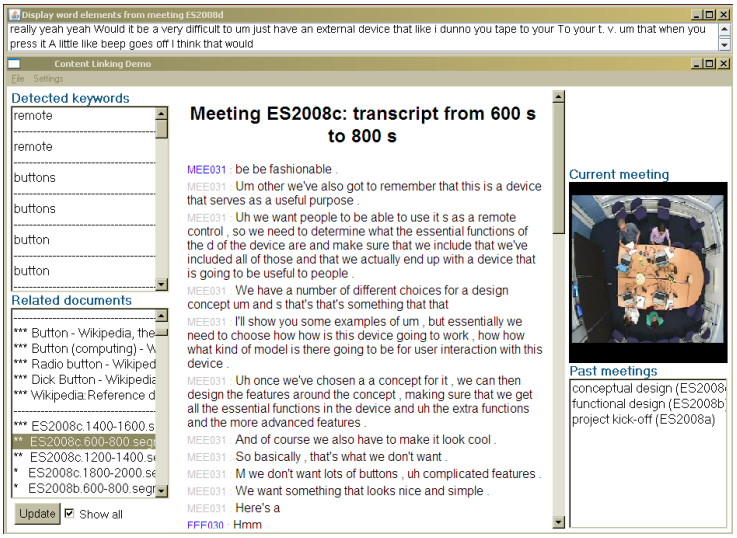

Figure 1: Snapshot of the ACLD Wide-screen UI, showing a fragment of a past meeting that is retrieved among the list of relevant documents at the bottom left corner, and is open in the central frame.

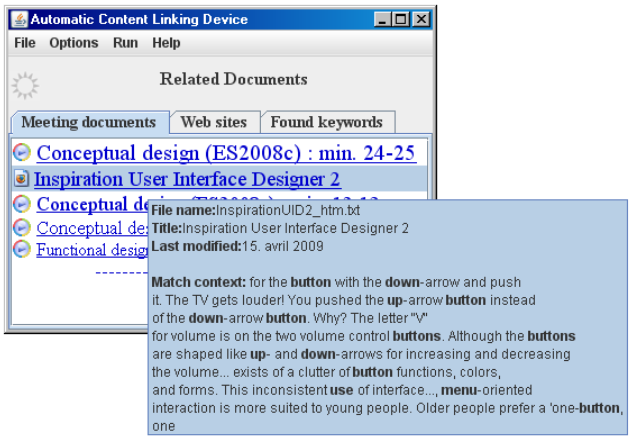

Figure 2: The ACLD Widget UI, showing a tab with the list of potentially relevant documents found by the system at a given moment, with explicit labels. Hovering over a label displays the metadata associated with the document, as well as the excerpts from it in which the keywords were found.

terfaces such as those shown in Figures 1 and 2. A number of modules are required to produce and exchange data, to ensure the following functionalities:

1. Prepare a database of "pseudo-documents", i.e. chunks of previous meetings in which media and annotations are aligned, with their associated (true) documents.

2. Prepare a query to this database, derived from the ongoing conversation through ASR or keyword spotting, possibly emphasizing certain keywords.

3. Execute the query, retrieve the results, and apply a context model to integrate them with previous results, in order to smooth variation in time of the document set.

4. Display the results and provide through them access to the past data, opening a meeting browser on fragments of past meetings, or a document visualizer. The system provides two main types of user interfaces: the Wide-screen UI (Figure 1), and the Widget UI (Figure 2), which is less intrusive but less informative as well. A modular UI which allows users to switch between the two types was also implemented.

4. ACKNOWLEDGMENTS

This work was supported by the EU IST Program, through the AMIDA IP FP6-0033812, and by the Swiss National Science Foundation, through the IM2 NCCR.

5. ADDITIONAL AUTHORS

Additional authors: Erik Boertjes (TNO, The Netherlands, email: `erik.boertjes@tno.nl`), Jean Carletta (HCRC, University of Edinburgh, UK, email: `J.Carletta@ed.ac.uk`), Sandro Castronovo (DFKI, Germany, email: `sandro.castronovo@dfki.de`), Michal Fapso (Brno Institute of Technology, Czech Republic, email: `ifapso@fit.vutbr.cz`), Mike Flynn (Idiap Research Institute, Switzerland, email: `mike.flynn@idiap.ch`), Alexandre Nanchen (Idiap Research Institute, Switzerland, email: `alexandre.nanchen@idiap.ch`), Theresa Wilson (HCRC, University of Edinburgh, UK, email: `twilson@inf.ed.ac.uk`), Joost de Wit (TNO, The Netherlands, email: `joost.dewit@tno.nl`), & Majid Yazdani (Idiap Research Institute, email: `majid.yazdani@idiap.ch`).

6. REFERENCES

[1] J. Carletta. Unleashing the killer corpus: experiences in creating the multi-everything AMI Meeting Corpus. *Language Resources and Evaluation Journal*, 41(2):181–190, 2007.

[2] P. E. Hart and J. Graham. Query-free information retrieval. *IEEE Expert: Intelligent Systems and Their Applications*, 12(5):32–37, 1997.

[3] B. J. Rhodes and P. Maes. Just-in-time information retrieval agents. *IBM Systems Journal*, 39(3-4):685–704, 2000.