TKK Reports in Information and Computer Science
Espoo 2008 TKK-ICS-R13

DEFINITION OF ENRICHED RELEVANCE FEEDBACK IN PICSOM

Deliverable D1.3.1 of FP7 project n° 216529 PinView

Jorma Laaksonen

TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY
TECHNISCHE UNIVERSITÄT HELSINKI
UNIVERSITE DE TECHNOLOGIE D'HELSINKI

TKK Reports in Information and Computer Science
Espoo 2008 TKK-ICS-R13

DEFINITION OF ENRICHED RELEVANCE FEEDBACK IN PICSOM

Deliverable D1.3.1 of FP7 project n° 216529 PinView

Jorma Laaksonen

Helsinki University of Technology
Faculty of Information and Natural Sciences
Department of Information and Computer Science

Teknillinen korkeakoulu
Informaatio- ja luonnontieteiden tiedekunta
Tietojenkäsittelytieteen laitos

Distribution:
Helsinki University of Technology
Faculty of Information and Natural Sciences
Department of Information and Computer Science
P.O.Box 5400
FI-02015 TKK
FINLAND
URL: http://ics.tkk.fi
Tel. +358 9 451 1
Fax +358 9 451 3369
E-mail: series@ics.tkk.fi

ISBN 978-951-22-9675-0 (Print)
ISBN 978-951-22-9676-7 (Online)
ISSN 1797-5034 (Print)
ISSN 1797-5042 (Online)
URL: http://lib.tkk.fi/Reports/2008/isbn9789512296767.pdf

TKK ICS
Espoo 2008

ABSTRACT: This report defines and implements communication principles and data formats for transferring enriched relevance feedback to the PicSOM content-based image retrieval system used in the PinView project. The modalities of enriched relevance feedback include recorded eye movements, pointer and keyboard events and audio including speech. The communication is based on the AJAX technology, where the client and server exchange XML formatted content by using the XMLHttpRequest method.

KEYWORDS: content-based image retrieval, enriched relevance feedback, file formats, data transfer, AJAX, XMLHttpRequest

ACKNOWLEDGEMENT: The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007–2013) under grant agreement n° 216529, Personal Information Navigator Adapting Through Viewing, PinView. All deliverables of the PinView project are available at `http://www.pinview.eu/` .

CONTENTS

1 OVERVIEW

The deliverable that constitutes the output of Task 1.3 *Definition of transport protocol for enriched feedback* of the *Personal Information Navigator Adapting Through Viewing*, PinView, project, funded by the European Community's Seventh Framework Programme under Grant Agreement n° 216529, consist of two parts: First, this report Deliverable D1.3.1 *Definition of enriched relevance feedback in PicSOM*. Second, a prototype Deliverable D1.3.2 *Implementation of enriched relevance feedback* that substantiates the specified protocol in the PicSOM content-based image retrieval (CBIR) system, developed at the Helsinki University of Technology (TKK). The report aims at defining in sufficient detail the data packet formats and interprocess communication methods needed in implementing the transfer of *enriched relevance feedback* from a web browser to a content-based image retrieval system.

In the PinView project, asynchronous communication mode between the information client and server is necessitated by the need to develop *proactive* and intelligent machine behaviour based on *implicit* user actions rather than *explicit* commands. The modalities of user interaction that can be used for extracting enriched relevance feedback include *eye movements*, *pointer movements and events*, *keyboard events* and *audio* including speech. The asynchronous transfer of recorded computer–human interactions from the browser to the search server is implemented by using the XMLHttpRequest protocol defined by the World Wide Web Consortium (W3C) and often used in implementing *Asynchronous JavaScript and XML* (AJAX) type of asynchronous content updates in the web.

The adopted file formats are based on the XML data format specified in the European Community's Sixth Framework Programme Network of Excellence project *COGAIN* for device-independent archiving and streaming of eye movement measurements. The COGAIN file format [1, 2] was found to be appropriate for being reused in the PinView project in the earlier Tasks 1.1 *Study of different forms of enriched feedback* [17] and 8.1 *Specification of information interfaces in PinView* [6].

The implementation of the transfer protocol and data formats has been substantiated within the PicSOM CBIR system developed at TKK since 1998. The implementation described in this report has been programmed during the summer and early autumn of 2008 in the Department of Information and Computer Science of TKK.

The source code of the PicSOM system has been written in the ANSI ISO/IEC 14882:1998 standard C++ programming language and can be compiled on various hardware platforms and in many operating systems. The correct operation of the prototype has been verified in a PC computer equipped with an Intel Core2 Duo CPU, running Ubuntu 8.04 distribution of the Linux operating system and the free GNU gcc/g++ compiler version 4.2.3, and in an Intel Mac computer, running OS X 10.4.11 operating system and Xcode compiler version 2.4.

The work presented in this report will be continued in the PinView Tasks 1.4 *Implementation of pointer track feedback*, 1.5 *Implementation of point-and-speak feedback* and 8.4 *Study of proactive eye-movement-based user interface*.

2 INTRODUCTION

The main goal of this report is to define the necessary communication methods and data formats needed in implementing the transmission of *enriched relevance feedback* (ERF) from a web browser client to a content-based image retrieval (CBIR) server. These specifications have simultaneously been implemented in the PicSOM CBIR system[1] [10], developed at the Helsinki University of Technology (TKK). During the course of the PinView project[2], these communication methods will be utilised in implementing a prototype of a *proactive personal information navigator* that allows retrieval of visual information available in versatile image databases by using not only *explicit* but also *implicit* or *unintentional* search cues from the user. The modalities of user interaction concerned in PinView and in this report include *eye movements*, *pointer movements and events*, *keyboard events* and *audio* including speech.

Figure 1 illustrates the overall schematic diagram of a client–server based image retrieval system with multimodal user interaction. In the context of this report we will assume that the user inputs will be utilised as enriched relevance feedback to the image search server. The implementation of proactive retrieval behaviour necessitates the use of *asynchronous* data transfer between the client and the server. This report details how the asynchronous transfer of recorded computer–human interactions from the browser to the search server can be implemented by using the World Wide Web Consortium's (W3C)[3] XMLHttpRequest protocol[4]. The XMLHttpRequest protocol has recently been employed extensively for implementing *Asynchronous JavaScript and XML* (AJAX)[5] type of asynchronous content updates in diverse *Rich Internet Applications* (RIAs)[6]. AJAX technologies can be used to implement web applications that communicate with some information server in the background and modify the content displayed to the user by following instructions sent by the server.

The file formats adopted in this report for transmitting the enriched relevance feedback data from the client to the server are based on the Extensible Markup Language (XML)[7]-based data format specified in the European Community's Sixth

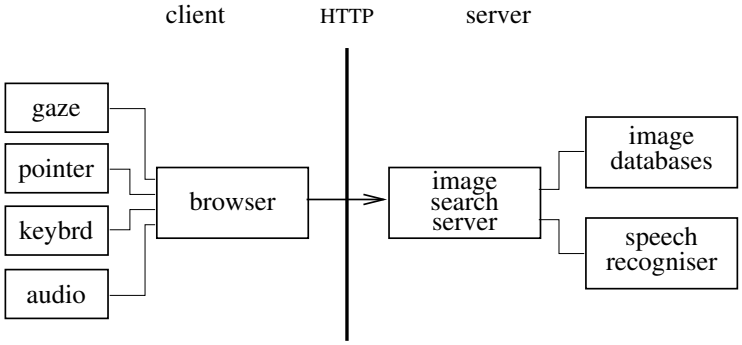

Figure 1: Block diagram of a system capable of transferring enriched relevance feedback from a browser client to a content-based image search server.

[1] http://www.cis.hut.fi/picsom
[2] http://www.pinview.eu/
[3] http://www.w3.org/
[4] http://www.w3.org/TR/XMLHttpRequest/
[5] http://www.adaptivepath.com/ideas/essays/archives/000385.php
[6] http://en.wikipedia.org/wiki/Rich_Internet_application
[7] http://www.w3.org/XML/

Framework Programme Network of Excellence project COGAIN[8] for device-independent archiving and streaming of eye movement measurements. The COGAIN file format [1, 2] was found to be appropriate for being reused in the PinView project in PinView's earlier Tasks 1.1 *Study of different forms of enriched feedback* [17] and 8.1 *Specification of information interfaces in PinView* [6].

The rest of this report is organised as follows. In Section 3, we begin by studying the general operating principles of a client–server system that implements asynchronous relevance feedback from the client to the server combined with the possibility for the server to proactively update the client's displayed content. In Section 4, the elements of the asynchronous communication are specified for the purpose of transmitting enriched relevance feedback to the server. In Section 5, specifications for packeting on-line measurement data of four user interaction modalities (eye movements, pointer events, keyboard events and audio) are presented in the form of usage examples. In Section 6, it is shown how the implementation of the specified data communications can be examined in the PicSOM CBIR system. Finally in Section 7, we present conclusions and a discussion on the related future work in the PinView project.

3 OPERATING PRINCIPLES

In this section we will first address the general questions regarding the implementation of asynchronous relevance feedback from a web browser to a content-based image retrieval system. The asynchronous communication mode is necessitated by the central goal of the PinView project to develop *proactive* and intelligent machine behaviour based on *implicit* user actions rather than *explicit* commands.

3.1 Client–server model

Figure 2 shows the messaging diagram of synchronous and asynchronous communications between an image database server and a browser client. The image retrieval session is initiated by the client requesting the server to present some visual content for inspection. The server returns with a conventional synchronous HTTP action a set of images to the browser, which then presents them to the user. Together with the XML/HTML page containing the images, the server can also specify an URL from its own URL space where the client can send asynchronous user interaction data. To our knowledge, this is a novel idea not used in any existing content-based image retrieval systems.

The message types transferred between the server and the client have been identified as **A**, **B**, **C** and **D** in Figure 2. Their exact roles in the communication of enriched relevance feedback can be detailed as follows:

Message A is a standard HTTP GET or POST request by which the client sends the server an image retrieval query, to which the server should respond with a new XML/HTML page containing the images and possible other content to be displayed to the user.

Message B includes the XML/HTML content provided by the server as a synchronous response to Message A. The only difference compared to conventional HTML content is that the response can contain a series of XML/HTML `<meta>` tags for specifying the browser operations related to the enriched rele-

[8] `http://www.cogain.org/`

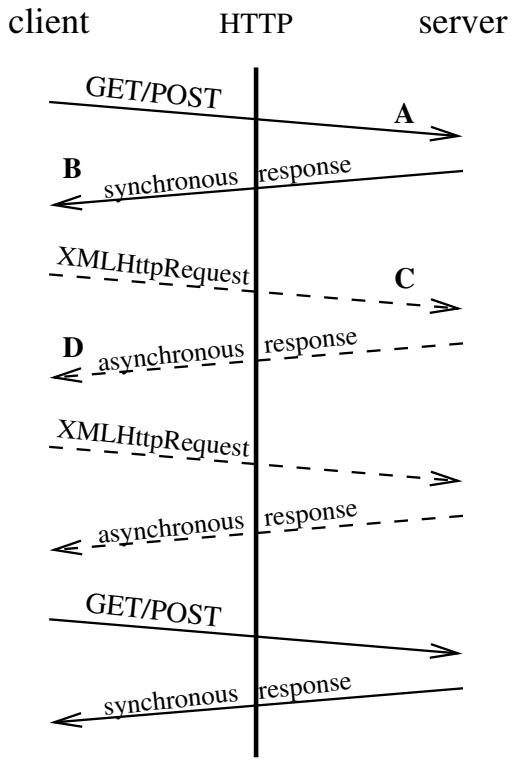

Figure 2: Message exchange diagram of synchronous and asynchronous communications between the image search client and the content-based image retrieval server.

vance feedback. Most importantly, the URLs where to send the asynchronous relevance feedback data are determined.

Message C is an asynchronous packet of enriched relevance feedback data recorded from the user interactions and sent by the browser to the search engine. For this step to succeed it is necessary that the previous synchronous Message B has contained proper URL addresses that can receive the data. The asynchronous communication is implemented by using the XMLHttpRequest protocol.

Message D is an asynchronous response packet from the server to the client. In most cases it will be just an acknowledgement of the received the data. However, it may also contain AJAX type instructions how to update the XML/HTML Document Object Model (DOM)[9] object received in Message B that specifies the content displayed by the browser to the user. This topic is out of the scope of this report and will be specified in later stages of the implementation of the PinView project.

3.2 Client–collector model

The same communication principles can also be used in a setting where the image database server is either not receiving or is receiving but not using the asynchronous user interaction data from the browser client during the query session. The latter case may be useful for collecting user interaction data for off-line studies. The former case is depicted in Figure 3, where there is a second HTTP server that actually receives the user interaction data and is called a *collector*. In this setting either:

[9] `http://www.w3.org/DOM/`

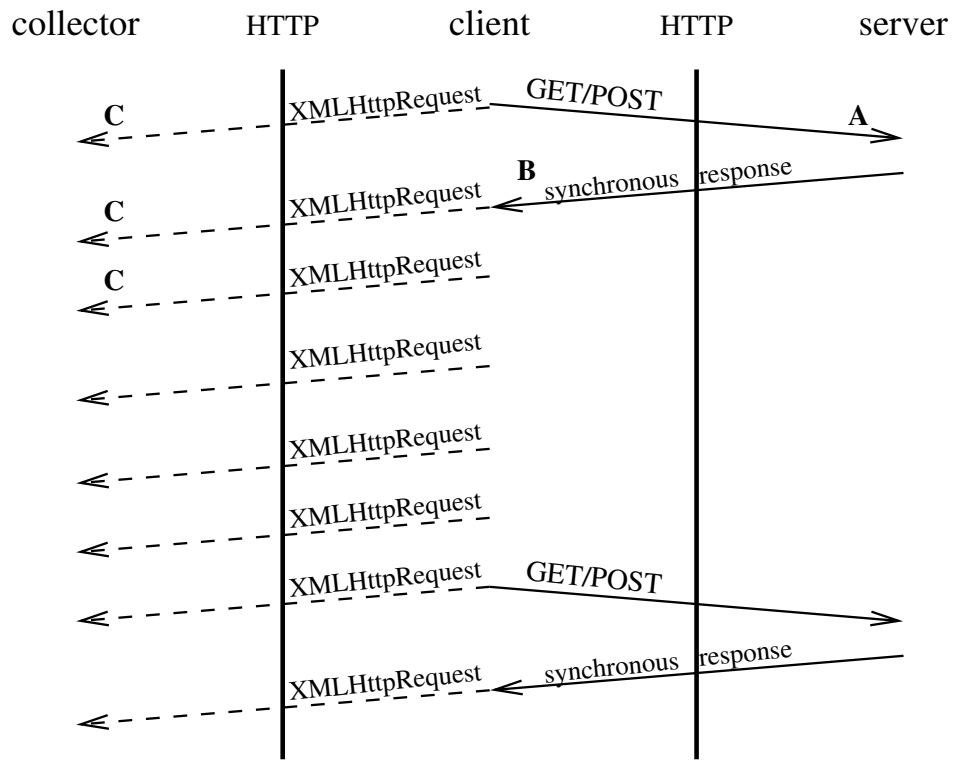

Figure 3: Message exchange diagram of synchronous and asynchronous communications.

- the information server specifies some other HTTP server than itself as the collector, or
- the information server either does not specify receiver for the asynchronous data transfers at all, or
- the user has explicitly instructed the browser client not to obey the data collection instructions from the information server.

In the last two cases the URL of the collector needs to be manually specified in the browser's configuration. Now the Messages A, B and C are analogous to those described in Section 3.1. As seen in Figure 3, the client will also inform the collector of its Message A and Message B type communications with the server by sending C messages, whose exact content will be defined later. The Message D type of communication is missing as the collector will in general be unable to provide asynchronous and proactive updates to the client. Consequently, the collector's responses to the XMLHttpRequests will be mere acknowledgement messages.

The client–collector model will be of only limited interest in the PinView project as long as the PicSOM CBIR system and publicly available image collections are being used in the user experiment. However, if some proprietary third party image search system or image collections were used, then the PicSOM system could act as the collector. It could thus record the user interactions for later analysis and possibly also provide additional retrieval results from its own databases.

3.3 Requirements for the browser client

The user's web browser will in general need to be equipped with a special plug-in or extension that is capable of interpreting the `<meta>` tags in the server's messages.

Then the browser extension needs to record data of the specified user interaction modalities, store it in proper XML formatted packets. Finally, it has to communicate the recorded data over the XMLHttpRequest protocol to the search server. The XMLHttpRequest protocol is already available in all modern web browsers including Internet Explorer, Firefox, Opera and Safari.

For implementing the proactive search interface, the browser client has to be also able to update the content of the displayed image pages by following the asynchronous instructions sent by the search server. How this is implemented in practice, will be a topic for future work.

3.4 Requirements for the search server

The content-based image retrieval server has to implement a mechanism for receiving the enriched relevance feedback from XMLHttpRequest messages. This will in general be quite simple as a functioning web-based CBIR system will in any case contain a subsystem capable of HTTP communication and the XMLHttpRequest messages do not differ significantly from other HTTP communications. The most important requirement is that the server is able to associate the input data with the correct ongoing query instance in its memory. For fulfilling this the server–client architecture will have two lines of options:

1. the XMLHttpRequest URL will specify the identity of the query, or

2. the XMLHttpRequest URL will be generic and the XMLHttpRequest message will specify the query identity.

The current implementation in the PicSOM CBIR system follows option 1.

For providing retrieval results proactively to the search client, the CBIR server needs to have a subsystem that is able to make use of the enriched relevance feedback data received. How this can implemented in practice is the central research topic of the PinView project and is not addressed further in this report.

4 SPECIFICATION OF ASYNCHRONOUS COMMUNICATION

The asynchronous communication between the browser client and the search server basically consists of three stages.

- First, with a Message B in Figure 2, the server declares its ability to receive asynchronous packets containing enriched relevance feedback data. This step will be described below in Section 4.1.

- Second, with messages of type Message C in Figure 2, the client sends ERF recordings asynchronously to the server. In these packets the client specifies the context in which the feedback data has been recorded. The general form of these packets will be described in Section 4.2 and later exemplified in Section 5.

- Third, with messages of type Message D in Figure 2, the server sends updates to the content and interface displayed by the client. The format of these messages will be specified at a later stage.

```
<?xml version="1.0"?>
...
  <meta name="erf/TYPE"
        content="http://www.pinview.eu/ajax.cgi"
        scheme="pinview/1.0"/>
...
```

Figure 4: Header of an XML/HTML file that specifies which forms of enriched relevance feedback should be collected by the browser client and sent to the search server for processing.

4.1 Specification of ERF data receiver

Figure 4 shows how the image search server specifies an URL address where the browser should send the recorded ERF data packets asynchronously. In practice that URL should refer back to the server's own address space and the server should be enabled to process XMLHttpRequests sent in that address.

The details of the `<meta>` element are as follows:

- In an HTML document the `<meta>` element is inside the `<head>` element which in turn is inside the `<html>` element.
- In the `name` attribute, TYPE is a combination of words `gaze`, `audio`, `pointer` and `keyboard`, separated with plus (+) signs. If the `/TYPE` specifier has been omitted or equals `/*`, that is interpreted as if the specification had been given as `/gaze+audio+pointer+keyboard`.
- There can be more than one `<meta name="erf/TYPE"/>` specification in the document. In that case the matching enriched relevance feedback modality will be sent to *all* specified locations.
- The `content` attribute specifies the actual URL that can be used as the server address in the XMLHttpRequest calls made by the browser.
- The `scheme` attribute provides a rudimentary mechanism for the client to select data formats and transfer models compatible with the server. In the context of this report, the scheme specification `pinview/1.0` will mean that the data packets need to be sent to the server by using the XMLHttpRequest protocol with the POST method and that their internal contents need to be of the formats of the examples in Section 5.

4.2 Common requirements for the ERF data packets

All enriched relevance feedback data packets sent to the search engine or data collector are in XML format and have some properties in common. These are exemplified in Figure 5.

- The root element of the packets will follow the convention of COGAIN's streamed eye movement recording data and be `<stream.start>`.
- There will be an `<erf>` element that refers to the URL of the displayed web page, which forms the context where the relevance feedback was recorded. The `scheme` attribute of the element is copied from the corresponding attribute in the `<meta>` element of the search engine specification. In addition,

```
<?xml version="1.0"?>
<stream.start>
  <erf href="http://www.pinview.eu/" type="TYPE"
       scheme="pinview/1.0"/>
  <data>
  ...
    <timestamp>Sat, 20 Sep 2008 07:09:42.173 GMT</timestamp>
    <duration>35</duration>
  ...
  </data>
</stream.start>
```

Figure 5: Common elements of the XML file format used for transferring ERF data from the browser client to the image search server.

the element specifies with the `type` attribute the type of the relevance feedback data contained in the rest of the packet. The type is a plus (+) sign separated combination of keywords `gaze`, `pointer`, `keyboard` and `audio`.

- The actual data packets will be wrapped inside a series of `<data>` XML elements.
- All collected recordings and events will be timed with an absolute timestamp in Greenwich Mean Time (GMT). The format of the `<timestamp>` element follows that specified in RFC1123[10] and the HTTP standard RFC2616[11] Section 3.3.1 with one exception. For the current purposes the temporal resolution of one second will not be sufficient and therefore we added the possibility that the time specifier contains an additional decimal point (.) followed by any number of decimals.
- Various forms of recordings will also contain a `<duration>` element that specifies the duration of the event measured in milliseconds and presented without units. This notation follows that of the COGAIN data format.

5 SPECIFICATION OF DATA FORMATS

In this section, we specify the formats of the data packets used for transmitting the recorded enriched relevance feedback from the client side to the server. The specifications span four modalities, i.e. eye movement or gaze, pointer movement, keyboard event and audio data. The specifications are based on the COGAIN data format and its later extensions sketched in PinView Deliverable D8.1 *Specification of information interfaces in PinView*.

At the current stage, the nature of the specifications is still more descriptive than normative, i.e. usage examples have been provide instead of formal definitions. This is mainly because the actual software implementations of the client side are under development and the details of the data formats are subject to changes. Later, when the data structures mature, explicit XML Schema[12] definitions and validation procedures for them will be established in the framework of the PicSOM CBIR system.

[10] `http://www.ietf.org/rfc/rfc1123.txt`
[11] `http://www.ietf.org/rfc/rfc2616.txt`
[12] `http://www.w3.org/XML/Schema`

5.1 Eye movement data

Figure 6 shows an example of an XML file used to transfer eye movement data, including raw eye-wise measurements and locations and durations of gaze fixations.

In addition to the XML elements common to all ERF packets and described in Section 4.2, the eye movement data packets contain the following elements:

- The `<erf>` element's `type` attribute includes `gaze`.
- A `<screen>`, `<window>` or `<image>` element to specify the mapping reference of the eye coordinates to follow.
- A `<gaze>` element to further specify the origin of the data.
- A `<sample>` or `<fixation>` element to specify whether the data item is a raw measurement or a detected fixation. Both will contain a `<timestamp>` element, the `<fixation>` element additionally a `<duration>` element.
- The `<x>` and `<y>` coordinates relative to the specified mapping reference. These can be inside an `<eye>` element that specifies which eye is in question with `side="left"` or `side="right"`.

```
<?xml version="1.0"?>
<stream.start>
  <erf href="http://www.pinview.eu/" type="gaze"
       scheme="pinview/1.0"/>
  <data>
    <window>
      <gaze>
        <sample>
          <timestamp>Sat, 20 Sep 2008 07:09:42.123 GMT</timestamp>
          <eye side="left"><x>200</x><y>150</y></eye>
        </sample>
      </gaze>
    </window>
  </data>
  <data>
    <window>
      <gaze>
        <fixation>
          <timestamp>Sat, 20 Sep 2008 07:09:42.173 GMT</timestamp>
          <duration>35</duration><x>205</x><y>156</y>
        </fixation>
      </gaze>
    </window>
  </data>
</stream.start>
```

Figure 6: XML file format used for transferring eye movement data.

5.2 Pointer movement data

Figure 7 shows an example of an XML file used to transfer pointer movement data, including information on pointer button events.

In addition to the XML elements common to all ERF packets and described in Section 4.2, the pointer movement and event data packets contain the following elements:

- The `<erf>` element's `type` attribute includes `pointer`.
- A `<pointer>` element to further specify the origin of the data.
- A `<sample>` or `<click>` element to specify whether the data item is a free pointer movement or a detected button click, i.e. a button press and release sequence without intermediate pointer movement. Both will contain an element `<timestamp>` and the `<click>` element additionally a `<duration>` element.
- The `<x>` and `<y>` coordinates of the pointer.
- A `<buttons>` element contains a plus (+) sign separated list of pressed buttons such as `left`, `middle` or `right`.

```
<?xml version="1.0"?>
<stream.start>
  <erf href="http://www.pinview.eu" type="pointer"
       scheme="pinview/1.0"/>
  <data>
    <pointer>
      <sample>
        <timestamp>Sat, 20 Sep 2008 07:09:42.123 GMT</timestamp>
        <x>298</x><y>302</y>
        <buttons>left</buttons>
      </sample>
    </pointer>
  </data>
  <data>
    <pointer>
      <click>
        <timestamp>Sat, 20 Sep 2008 07:09:42.173 GMT</timestamp>
        <duration>350</duration>
        <x>190</x><y>133</y>
        <buttons>left</buttons>
      </click>
    </pointer>
  </data>
</stream.start>
```

Figure 7: XML file format used for transferring pointer movement and event data.

5.3 Keyboard event data

Figure 8 shows an example of an XML file used to transfer keyboard event data, including information on key names, movement directions and modifier keys.

In addition to the XML elements common to all ERF packets and described in Section 4.2, the keyboard event data packets contain the following elements:

- The `<erf>` element's `type` attribute includes `keyboard`.
- A `<keyboard>` element to further specify the origin of the data.
- A `<key>` element that contains a `<timestamp>` element.
- A `<name>` element to specify the name of the key in question.
- A `<direction>` element to specify the direction of the key movement, i.e. `down` or `up`.
- A `<modifiers>` element contains a plus (+) sign separated list of active modifier keys such as `shift`, `ctrl`, `alt` and `meta`.

```
<?xml version="1.0"?>
<stream.start>
  <erf href="http://www.pinview.eu" type="keyboard"
       scheme="pinview/1.0"/>
  <data>
    <keyboard>
      <key>
        <timestamp>Sat, 20 Sep 2008 07:09:42.123 GMT</timestamp>
        <name>x</name>
        <direction>down</direction>
        <modifiers>ctrl</modifiers>
      </key>
    </keyboard>
  </data>
</stream.start>
```

Figure 8: XML file format used for transferring keyboard event data.

5.4 Audio data

Figure 9 shows an example of an XML file used to transfer raw audio data from the client to the server for further processing, possibly including speech recognition.

In addition to the XML elements common to all ERF packets and described in Section 4.2, the eye movement data packets contain the following elements:

- The `<erf>` element's `type` attribute includes `audio`.
- An `<audio>` element to further specify the modality of the data.
- A `<messagetype>` element to specify the type of the packet's content: `AUDIO` for audio data, `AUDIO_END` for marking the end of an audio transfer, `RESET` for forcing reset of the speech recognition system, `RECOG` to identify recognition results provided by the speech recogniser and `RECOG_END` to identify end of recognition output.
- An `<urgent>` element, whose non-zero value identifies that the corresponding audio packet needs to be processed before any other queued audio packet.
- A `<waveform>` element that contains the actual binary data inside an XML CDATA section. Parameters such as `rate` in Herz, the `format` of samples and the number of `channels` can be specified. Currently the data needs to be monophonic signed 16-bit integer values sampled at 16 kHz. More advanced compressed waveform formats will be considered later if the audio bandwidth turns out to be a bottleneck.

```
<?xml version="1.0"?>
<stream.start>
  <erf href="http://www.pinview.eu" type="audio"
       scheme="pinview/1.0"/>
  <data>
    <audio>
      <timestamp>Sat, 20 Sep 2008 07:09:42.123 GMT</timestamp>
      <duration>30</duration>
      <messagetype>AUDIO</messagetype>
      <urgent>0</urgent>
      <waveform rate="16000" format="16bit" channels="1"><![CDATA[
       ... ... ... ... ... ... ]]></waveform>
   </audio>
  </data>
</stream.start>
```

Figure 9: XML file format used for transferring audio data.

6 IMPLEMENTATION

The asynchronous transfer of enriched relevance feedback defined in the previous section has been implemented in the PicSOM CBIR system. This section first gives a brief introduction to the PicSOM system and describes its compilation and invocation. Then examples are presented on how XML files containing packets of enriched relevance feedback data can be communicated to the server.

6.1 PicSOM

PicSOM[13] [9, 10] is a content-based information retrieval system developed at the Helsinki University of Technology since 1998, first in the Laboratory of Computer and Information Science and later in the Department of Information and Computer Science. The unique approach used in PicSOM is to have several Self-Organizing Maps (SOMs) [5] in parallel to index and determine the similarity of data objects. These parallel SOMs have been trained with separate data sets obtained by using different feature extraction algorithms on the same objects. So each SOM arranges the same objects differently, according to the particular multi-dimensional feature vectors used in its training.

PicSOM uses the principles of *query by example* [3] and *relevance feedback* [12, 11] in implementing iterative and interactive image retrieval. This means that the system shows the user a set of database objects, which the user then indicates as relevant or non-relevant to the current query, i.e. close to or far from what he is looking for. Based on this relevance feedback information, PicSOM modifies its internal parameters so that in the next round it will display objects that resemble those that had been marked as relevant. This is done by increasing the influence of those SOMs that give the most valuable similarity evaluation according to the current relevance feedback information. The user thus becomes an integral part of the query process, which can be seen as a form of supervised learning, where the user steers the system by providing feedback. A CBIR system implementing relevance feedback essentially tries to learn the optimal correspondence between the high-level human concepts and the low-level internal features used by the system.

The PicSOM CBIR system was initially designed to index and retrieve images only. Segmentation was introduced into PicSOM [16], and later we have used image segments in parallel with entire images to improve retrieval results [14]. This algorithm was then generalised to be used with multi-part objects such as web-pages containing images and links [13] and video retrieval [7, 15, 8].

6.2 Compilation

Assuming that one has the source code of the PicSOM CBIR system stored in a file named `~/picsom-0.11.tar.gz`, one can configure and compile the system. The procedure has been tested also in a Mac OS X system, but the following examples assume the program is compiled and executed in a Linux host called here as `linux-box`:

```
linux-box %> cd ~/picsom
linux-box %> tar xfz ~/picsom-0.11.tar.gz
linux-box %> cd picsom-0.11
linux-box %> ./configure
linux-box %> make
linux-box %> ln -s $PWD/js $PWD/xsl $PWD/xsd ~/picsom
```

[13] `http://www.cis.hut.fi/picsom`

6.3 Invocations

Provided that one has at least one image database stored in the PicSOM directory named `~/picsom/databases` in the format specified in [6], the PicSOM server process can be invoked for testing its operation in interactive content-based image retrieval:

```
linux-box %> src/picsom -Dajax -myport
```

The `-Dajax` switch instructs the system to display debugging information concerning the receiving of enriched relevance feedback. Even more debugging output from the HTTP operations can be obtained with the switch `-Dhttp=5`.

One can connect a web browser to the HTTP port 5600 of the PicSOM server by issuing:

```
any-host %> firefox http://linux-box:5600
```

6.4 Retrieval

After clicking the `databaselist` link one can select the database, features and retrieval algorithm from listed choices. During the retrieval session the user marks some of the query images as relevant and then presses the *Continue query* button.

Each round of the query will have a unique URL visible in the browser window. The XML content of the query can be accessed also by using the `wget` command line utility:

```
any-host %> wget http://linux-box:5600/query/Q:080930:010040:4730:0
```

6.5 ERF transfer

The `wget` utility can be used for simulating the XMLHttpRequest-based communication of enriched relevance feedback from the browser client. For example, simulated eye movement data can be submitted with the command:

```
any-host %> wget --post-file=gaze.xml \
                 http://linux-box:5600/ajax/Q:080930:010040:4730:0
```

Note, that the only difference between the synchronous and asynchronous call URLs is that the former have `/query/` in the path whereas the latter have `/ajax/`. For simplifying the AJAX calls, PicSOM will also recognise a special query identifier `Q:previous` that refers to the most recent synchronous query. The above command could thus be equivalently issued:

```
any-host %> wget --post-file=gaze.xml \
                 http://linux-box:5600/ajax/Q:previous
any-host %> cat Q:previous
<?xml version="1.0"?>
<?xml-stylesheet href="/picsom.xsl" title="Main style"
                 type="text/xsl"?>
<picsom:result
  xmlns:picsom="http://www.cis.hut.fi/picsom/ns">
  <picsom:ajaxresponse>Content not specified yet.
  </picsom:ajaxresponse>
</picsom:result>
```

As can be seen above, the current PicSOM implementation does not yet provide any useful response for the ERF input. It will nevertheless check the validity of the request and display in its debugging output a message showing the key properties of the stored data:

```
Query::ProcessAjaxRequest() : called
   href="http://www.pinview.eu"
   type="gaze"
 scheme="pinview/1.0"
   data="window"
 stored as ajax data #0
```

Similar behaviour can be seen also with simulated data packets of the other enriched relevance feedback modalities:

```
any-host %> wget --post-file=pointer.xml  \
                 http://linux-box:5600/ajax/Q:previous
any-host %> wget --post-file=keyboard.xml \
                 http://linux-box:5600/ajax/Q:previous
any-host %> wget --post-file=audio.xml    \
                 http://linux-box:5600/ajax/Q:previous
```

The debugging outputs from the PicSOM server is now:

```
Query::ProcessAjaxRequest() : called
   href="http://www.pinview.eu"
   type="pointer"
 scheme="pinview/1.0"
   data="pointer"
 stored as ajax data #1

Query::ProcessAjaxRequest() : called
   href="http://www.pinview.eu"
   type="keyboard"
 scheme="pinview/1.0"
   data="keyboard"
 stored as ajax data #2

Query::ProcessAjaxRequest() : called
   href="http://www.pinview.eu"
   type="audio"
 scheme="pinview/1.0"
   data="audio"
 stored as ajax data #3
```

As can be seen, in the current implementation the association of the ERF data with the query instance stored in the PicSOM server's memory is exclusively based on the URL address used in the (now simulated) XMLHttpRequest communication. The value of the `href` attribute of the `<erf>` element is thus ignored. This policy may change when true on-line XMLHttpRequest communication has been implemented in a browser client.

7 CONCLUSIONS

In this document we have specified data formats and an asynchronous data transfer protocol for communicating enriched relevance feedback information from a

browser client to a content-based image retrieval server. The protocol and data formats have simultaneously been implemented and tested with simulated data in the PicSOM CBIR system developed at TKK.

In the continuation of the PinView project we will obtain true on-line asynchronous enriched relevance feedback from users by employing browser applets or plug-ins to be developed in Tasks 1.4 *Implementation of pointer track feedback* and 1.5 *Implementation of point-and-speak feedback*. During the development of the plug-ins, some preliminary experiments will be required for selecting proper frequencies for the transmittal of the recorded data from the browser to the search server. In addition, it may be necessary to revise the data formats and the transfer protocol if, for example, the delays of the data transfers turn out to be a bottleneck of the operation. The integration of the speech recognition subsystem in the CBIR server, as depicted in Figure 1, is another research topic that can be addressed when the browser plug-in for audio input is available.

During the work it has turned out that somewhat related work has been carried out for example in MIT, where a research group has developed the Web-Accessible Multimodal Interfaces (WAMI) toolkit [4]. Their approach is tightly connected to the use of speech input and audio output in web applications and as such it does not seem to have directions to eye movement analysis. All in all, content-based image retrieval and extraction of implicit relevance feedback from eye movements still seem to make a combined research topic not thoroughly addressed by other research teams.

ACKNOWLEDGEMENTS

Collaborators in the PinView project, especially Teemu Hirsimäki, Markus Koskela and Mats Sjöberg of TKK and Bernhard Lackner of celum, are acknowledged for their valuable comments and contributions concerning the content of this report.

References

[1] Richard Bates, Howell Istance, and Oleg Spakov. Requirements for the common format of eye movement data. Communication by Gaze Interaction (COGAIN), IST-2003-511598: Deliverable 2.2., October 2005.

[2] Richard Bates and Oleg Spakov. Implementation of COGAIN gaze tracking standards. Communication by Gaze Interaction (COGAIN), IST-2003-511598: Deliverable 2.3., March 2006.

[3] N.-S. Chang and K.-S. Fu. Query-by-Pictorial-Example. *IEEE Transactions on Software Engineering*, 6(6):519–524, November 1980.

[4] Alexander Gruenstein, Ian McGraw, and Ibrahim Badr. The WAMI Toolkit for developing, deploying, and evaluating web-accessible multimodal interfaces. In *Proceedings of Tenth International Conference on Multimodal Interfaces (ICMI 2008)*, Chania, Greece, October 2008.

[5] Teuvo Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer-Verlag, Berlin, third edition, 2001.

[6] Markus Koskela and Jorma Laaksonen. Specification of information interfaces in PinView. PinView FP7-216529 Project Deliverable Report D8.1, June 2008. Available online at `http://www.pinview.eu/deliverables.php`.

[7] Markus Koskela, Jorma Laaksonen, Mats Sjöberg, and Hannes Muurinen. PicSOM experiments in TRECVID 2005. In *Proceedings of the TRECVID 2005 Workshop*, pages 262–270, Gaithersburg, MD, USA, November 2005. Available online at `http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html`.

[8] Markus Koskela, Mats Sjöberg, Ville Viitaniemi, Jorma Laaksonen, and Philip Prentis. PicSOM experiments in TRECVID 2007. In *Proceedings of the TRECVID 2007 Workshop*, Gaithersburg, MD, USA, November 2007. Available online at `http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html`.

[9] Jorma Laaksonen, Markus Koskela, Sami Laakso, and Erkki Oja. Self-organizing maps as a relevance feedback technique in content-based image retrieval. *Pattern Analysis & Applications*, 4(2+3):140–152, June 2001.

[10] Jorma Laaksonen, Markus Koskela, and Erkki Oja. PicSOM—Self-organizing image retrieval with MPEG-7 content descriptions. *IEEE Transactions on Neural Networks, Special Issue on Intelligent Multimedia Processing*, 13(4):841–853, July 2002.

[11] Yong Rui, Thomas S. Huang, Michael Ortega, and Sharad Mehrotra. Relevance feedback: A power tool in interactive content-based image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5):644–655, September 1998.

[12] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. Computer Science Series. McGraw-Hill, New York, 1983.

[13] Mats Sjöberg and Jorma Laaksonen. Content-based retrieval of web pages and other hierarchical objects with Self-Organizing Maps. In *Proceedings of 15th International Conference on Artificial Neural Networks (ICANN 2005)*, pages 841–846, Warsaw, Poland, September 2005. Available online at `http://dx.doi.org/10.1007/11550907_133`.

[14] Mats Sjöberg, Jorma Laaksonen, and Ville Viitaniemi. Using image segments in PicSOM CBIR system. In *Proceedings of 13th Scandinavian Conference on Image Analysis (SCIA 2003)*, volume 2749, pages 1106–1113, Halmstad, Sweden, June/July 2003. Springer Verlag. Available online at `http://www.springerlink.com/content/65xgpgqc9cjquqcl/`.

[15] Mats Sjöberg, Hannes Muurinen, Jorma Laaksonen, and Markus Koskela. PicSOM experiments in TRECVID 2006. In *Proceedings of the TRECVID 2006 Workshop*, Gaithersburg, MD, USA, November 2006. Available online at `http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html`.

[16] Ville Viitaniemi. Image segmentation in content-based image retrieval. Master's thesis, Laboratory of Computer and Information Science, Helsinki University of Technology, 2002.

[17] He Zhang, Markus Koskela, and Jorma Laaksonen. Forms of enriched relevance feedback. PinView FP7-216529 Project Deliverable Report D1.1, April 2008. Available online at `http://www.pinview.eu/deliverables.php`.

TKK REPORTS IN INFORMATION AND COMPUTER SCIENCE

TKK-ICS-R3 Jussi Lahtinen
Model Checking Timed Safety Instrumented Systems. June 2008.

TKK-ICS-R4 Jani Lampinen
Interface Specification Methods for Software Components. June 2008.

TKK-ICS-R5 Matti Koskimies
Applying Model Checking to Analysing Safety Instrumented Systems. June 2008.

TKK-ICS-R6 Alexander Ilin, Tapani Raiko
Practical Approaches to Principal Component Analysis in the Presence of Missing Values. June 2008.

TKK-ICS-R7 Kai Puolamäki, Samuel Kaski
Bayesian Solutions to the Label Switching Problem. June 2008.

TKK-ICS-R8 Abhishek Tripathi, Arto Klami, Samuel Kaski
Using Dependencies to Pair Samples for Multi-View Learning. October 2008.

TKK-ICS-R9 Elia Liitiäinen, Francesco Corona, Amaury Lendasse
A Boundary Corrected Expansion of the Moments of Nearest Neighbor Distributions. October 2008.

TKK-ICS-R10 He Zhang, Markus Koskela, Jorma Laaksonen
Report on forms of enriched relevance feedback. November 2008.

TKK-ICS-R11 Ville Viitaniemi, Jorma Laaksonen
Evaluation of pointer click relevance feedback in PicSOM. November 2008.

TKK-ICS-R12 Markus Koskela, Jorma Laaksonen
Specification of information interfaces in PinView. November 2008.

ISBN 978-951-22-9675-0 (Print)
ISBN 978-951-22-9676-7 (Online)
ISSN 1797-5034 (Print)
ISSN 1797-5042 (Online)