TRECVID 2004
Experiments at MediaTeam Oulu

Mika Rautiainen[†], **Matti Hosio**[†], **Ilkka Hanski**[†], **Matti Varanka**[†], **Jukka Kortelainen**[†], **Timo Ojala and Tapio Seppänen**[†]

[†] MediaTeam Oulu

P.O.BOX 4500, FIN-90014 University of Oulu, Finland

{firstname.lastname@ee.oulu.fi} discard umlauts

Abstract

University of Oulu's MediaTeam research group participated in manual and interactive search tasks at TRECVID 2004 evaluation. Our experiments in manual search task evaluated the effect of using three different semantic levels of video shot similarity for retrieval: visual, concept and text. Where the visual search engine is based on computational low-level feature vector distances, text search engine matches the spoken words that are extracted from the audio track using speech recognition. Linear weights are defined to the search engines based on previous experiments with the TRECVID development database. The benefit of correcting ASR transcript errors with closed captions information as well as the significance of text query expansion was evaluated in the 2004 experiments. The role of the different search engines was also considered in the experimental setup.

Some improvements were implemented to the 2003 version of our video browser user interface in order to make interactive browsing more efficient in finding relevant video clips. The enhancements to the user interaction elements were composed of several quick launch buttons for fast shot playback, navigational operations and picking relevant shots into result set. Interface also logged user's browsing history to provide possibility to reverse the navigational steps. Showing more temporally grouped shots per screen increased the amount of visible information in the browser.

The experimental results for interactive search task were obtained by conducting a user experiment of twelve people with two system configurations: browsing by (I) ASR text-based features only (every computation of similarity between two shots was based on ASR descriptions) or (II) visual and ASR text-based features were combined. The interactive results using ASR-based features were on par or better than the results using combinations of the visual and ASR features. This paper gives an overview of the developed system and summarises the results.

1 Introduction

This paper first describes the system components (search engines) of the Video Browsing and Retrieval System (VIRE) as they were used in 2004 experiments. Following sections give details about the experiments in manual and interactive search tasks.

2 Manual and Interactive Search

2.1 Video Browsing and Retrieval System

The video browsing and retrieval system VIRE was used in manual and interactive search experiments. System is constructed using J2SE, QuickTime for Java and MySQL JDBC software components. Principally, the system consists of a server and client(s); the server provides query services for three search engines and formulates the final result set by fusing the sub-results from the independent engines. Client software has two views, one for constructing video queries manually and another for browsing the video shot database. References to media data and additional meta-information are stored in MySQL-database. Actual video data is stored on a network file system. The playback of video files and shots does not utilize any intermediary streaming service. The browser uses cluster-temporal browsing [9] to display dynamically relevant database content to the user.

VIRE server creates a search using three search engines (see Fig.1). Lowest-level engine computes visual similarity between two video objects and is typically used in a content-based example search. Concept search engine uses higher-level features extracted from

low-level features by pre-trained semantic concept detectors. Text search is based on automatic speech recognition transcripts, which makes the text search engine useful tool especially in news video material where spoken content is a significant part of the news media structure.

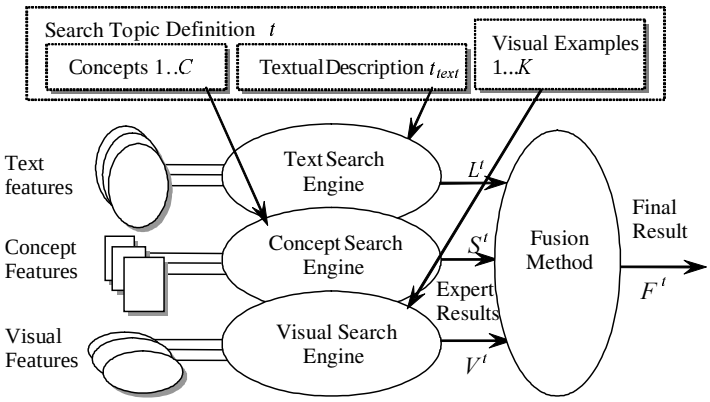


Figure 1. The VIRE test system used in TRECVID experiments

Visual Search Engine

Generic features are measured from the physical properties of the visual video data. Generic feature vectors are compared using a geometric distance that measures dissimilarities between two vectors. To compute the geometric dissimilarities of the described feature vectors, we have used L1 norm. Each feature vector is normalized prior computing the dissimilarity. Several features can be used simultaneously in search to compute overall similarity of two objects. The results of individual feature queries (ranked lists of database shots) are combined with late fusion technique to provide a final ranked set of most similar video shots. The fusion of different visual features is realized with a straightforward sum of ranks with equal weights. The benefit of using rank combination is fair when two features have very different dimensionalities. [10]

The visual similarity is constructed from the two physical properties of a shot: (I) *Color* is the most widely used content property in CBVR research. Similarity by color gives initially very good perceptual correspondence between two color images that are small or short of details. After visual content is reviewed in detail, other properties for similarity emerge from the details. (II) *Structure* of the edges in a visual imagery is a strong cue for many computer vision applications, such as classification city and landscape images or segregating natural and non-natural objects. This can also provide invaluable in queries where statistical color information is insufficient in describing the main properties of an image. In some occasions it can be the only property that can reliably describe the content of a sequence, a fast-paced action would be an example of such. Following features have been used in our experiments: Temporal Color Correlogram (TCC) and Temporal Gradient Correlogram (TGC). These features are computed from a sequence of shot frames. A closer description of the features follows.

Temporal Color Correlogram (TCC). Color properties of a video shot were obtained into a Temporal Color Correlogram (TCC) feature. Its efficiency against traditional static color descriptors has been indicated in [1][2]. TCC captures the correlation of HSV color pixel values in spatio-temporal neighborhoods. The feature approximates temporal dispersion or congregation of uniform color regions within a video sequence. Traditionally static color features do not consider temporal dynamics as they are often computed from a single keyframe. TCC is computed by sampling 20 video frames evenly over a bounded video sequence. The details about the computational parameters such as color space quantization and spatial constraints are described in [2].

Temporal Gradient Correlogram (TGC). Temporal Gradient Correlogram (TGC) feature computes local correlations of specific edge orientations producing an autocorrelogram, whose elements correspond to probabilities of edge directions occurring at particular spatial distances. Similar to TCC, feature vector is computed over 20 video frames sampled evenly over the duration of bounded video sequence. Due to shot sampling, autocorrelogram is affected by the temporal changes in spatial edge orientations. From each sample frame, the edge orientations (obtained with Prewitt kernels) are quantized into four segments depending on their orientation being horizontal, vertical or either of the diagonal directions. The details of the feature and the used parameters can be found from the study utilizing it in the detection of city/landscape scenes and people from video shots [6].

Concept Search Engine

Semantic features are lexically defined concepts that exist in a shot with a certain confidence value. Trained semantic concept detectors are used to create a series of confidences for each video shot. Our search system utilizes a set of concept confidences as a shot feature that is being compared against other shots' confidence values. One problem with floating confidence values is that they are generated with an algorithm that attempts to find out the degree in which the concept in question truly exists in the shot. Therefore the confidence value often starts to fail in some threshold value and generates non-relevant results for a concept-query. Our system uses 100 semantic concept filters from IBM VideoDIG project [11]. The IBM filters have created 100 ordered lists with a size of 2000 shots. A list is generated from the shots in test video database and the ranking is based on largest detected confidences for each concept. The concept detection lists are utilized in concept search engine, which computes overall concept similarity between two shots.

Manual concept query is based on user-defined set of concepts that has been attributed as relevant for the query. In order to achieve the overall concept similarity, confidences to every query concept are obtained as a rank value in the concept list. If the database item does not appear in the ordered concept list of 2000 shots, the concept is defined as indifferent for that item as there is no evidence whether a concept can be found from it. It is significant to note that the concept lists are not complete and define only partially the concept search space. This has a significant degrading impact on the recall of the concept search engine. Also, the list of 2000 shots does not take into account the frequency of occurrence. For some uncommon concepts, the frequency of occurrence is far less than 2000, perhaps by the magnitude of hundredths. On the other hand the more frequent concepts may occur in much more than 2000 shots. The dissimilarity of a query concept and a database item equals to item's confidence rank on that concept. Finally, after computing dissimilarities (confidence ranks) for every query concept, overall dissimilarity between the query concept list and a database item is computed as the sum of individual dissimilarity values. Final result list is sorted by the overall dissimilarities and is outputted as the result for concept query.

Over 100 semantic concepts were incorporated in our conceptual search. The full list of features can be found from [11]. Some samples from the beginning of the concept list: Addressing, Airplane, Airplane_Landing, Airplane_Takeoff, Animal, Baseball, Basketball, Beach, Bicycle, Bill_Clinton, Boat, Bridge, Briefing_Room_Setting, Building, Bus, Car, Car_Crash, Cartoon, Chair, Cheering, Chicken, Cityscape, Clapping, Classroom_Setting etc. Semantic search engine was used in the TRECVID 2004 manual search experiments, specifically in run `OUMT_M10`.

Text Search Engine

The ingredients for text indexes were automatic speech recognition (abbreviated as ASR) and closed caption (CC) transcripts. ASR text was produced at LIMSI [5] by running news audio through program, which automatically converted spoken audio to text. The speech was segmented person-wise (i.e. the change of the talking person was identified in the ASR transcripts). Each word in a speech segment was marked separately with proper time markers and the confidence value that denoted the accuracy of the recognition.

Another text source was the closed caption text that was supplied by organizer (NIST) and contained the actually spoken text written by a person. Therefore the text in it was accurate but contained no time markers for the word appearance. The incorporation of CC text with ASR would give a more accurate word database. The algorithm of Ratcliff and Metzener compares two files and marks the segments containing the exactly same lines [12]. The algorithm was implemented in Python programming language and a script was developed which compared ASR and CC words and replaced every differing ASR text with CC equivalent. If there were more CC words than ASR words, the time boundaries from ASR text had to be interpolated so that CC words fits to the ASR time domain.

The ASR transcripts were indexed into a database treating each white-space delimited token as a word. A stop word list was used to exclude grammatical and otherwise non-discriminating words that would have led to poor resolution. Remaining words were then stemmed using the Porter stemming algorithm [4].

Available speaker segmentation helped to create better contextual organization for the index and whip up the temporal search capabilities. This was achieved by expanding the inverted text index (appearances of the indexed terms in shots) into neighbouring shots until the boundaries of speaker segments were reached. Due to this, every shot became topically connected to its neighbours, presuming that the speaker would not change his/her semantic context during speech. Moreover, the temporal distance from the

original term appearance prioritised the neighbouring shots based on how far the shot was from the exact keyword shot. The sorting criterion was a priority value that was assigned to each shot. Priority values with zero is the highest possible (denoting that shot has the searched word) and then the next shot in the future would obtain the priority 1, next shot 3 and so on until the speech segment boundary is met. The priority of 2 was always assigned to the first shot behind the exact hit. Finally, the remaining shots before the exact hit were prioritized with values less than the last one in the speech segment. If there was more than one shot with exact hit, the highest possible priority was assigned to every one of them. In addition, shots of the surrounding speech segments before and after the matched segment were prioritized with lower values. This gives small weight to the shots containing neighboring speakers. It is based on the assumption that two news anchors are somewhat likely to speak about the same news context, although typically the news story segment boundary is at the point where speaker change occurs.

To compute the matching score between database shots and a query term, we used prioritised ranking combined with weighed term frequency score.

$$L(queryterm, s) = 0.2 \cdot \frac{\log(t+1)}{\log(dl+1)} * \log(\frac{N}{m}) + e^{-B\frac{j}{J}} \quad \textbf{(1)}$$

where s is the database shot to be scored, t is the number of matching words in the shot s, dl is the amount of all words in the text index (stop words are excluded), N is the number of shots in the test database and m is the amount of matching shots containing the searched word, B time constant, j the index (position) of s in the list of matching shots that is ordered by priority values, J is the size of the priority ordered list of shots. The given query words were stemmed to remove any suffixes. The scores were computed independently for every query term. The final scoring for the shot s was the sum of individual scores. Finally, the result list was sorted based on the final scores.

Since lexical similarity was also a search feature in the cluster-temporal browser, we constructed an example-based text search that used ASR text from the example shot as a source for query. ASR text was prepared using stop word removal and stemming. A list of lexically similar shots was then created using previously described scoring scheme.

Feature Fusion

In order to construct a final ranked list of most similar shots, VIRE system has to formulate sub-queries for the search engines. User can select whether he/she wants to retrieve shots based on all or some of the semantic search engines. For example, user can define manually following type of query description t:

- Text: '`Find shots of people moving a stretcher`'
- Concepts: `PEOPLE, WALKING, CAR, HUMAN`
- Visual: example shots of people moving a stretcher

The sub-results can be considered as votes of individual feature 'experts' e. To make a fusion of these feature lists, we use a Borda count [7] variant. Following formulas describe the fusion procedure

$$f^t(n) = sum(\frac{w_v \cdot v^t(n)}{V^t_{\max}}, \frac{w_s \cdot s^t(n)}{S^t_{\max}}, \frac{w_l \cdot l^t(n)}{L^t_{\max}}) \quad \textbf{(2)}$$

$$F^t = \left\lfloor sort\{f^t(1),...,f^t(N)\} \right\rfloor_X \quad \textbf{(3)}$$

where $f^t(n)$ = overall rank of a result shot n to the search topic definition t

$v^t(n), s^t(n), l^t(n)$ = rank of a result n by independent search engines

w_v, w_s, w_l = weights of the search engines

$V^t_{\max}, S^t_{\max}, L^t_{\max}$ = last rank of the independent result lists

F^t = Final ranked set of results for the search topic t

$[\ ]_X$ = X top-ranked items in a sorted list, where X equals to 1000 for TRECVID 2004 search experiments

Manual Query Interface

Manual query interface gives tools to define query attributes for the search engines. Changing a topic from the menu launches a timer and loads new example shots/images from the topic description files. User can select any combination of the topic examples for visual search and select the low-level features individually for each example. From the list of 100 semantic concepts, user can set any configuration of semantic features. Textual query terms are created by typing words to a text box. After user has enabled the desired engines for the search and selected the appropriate attributes for every selected engine a search is submitted to the VIRE server. Server distributes the query definitions to the respective search engines. When the result shots for the query have arrived, user can select any interesting shot from the result set as a start point for browsing with the browsing interface.

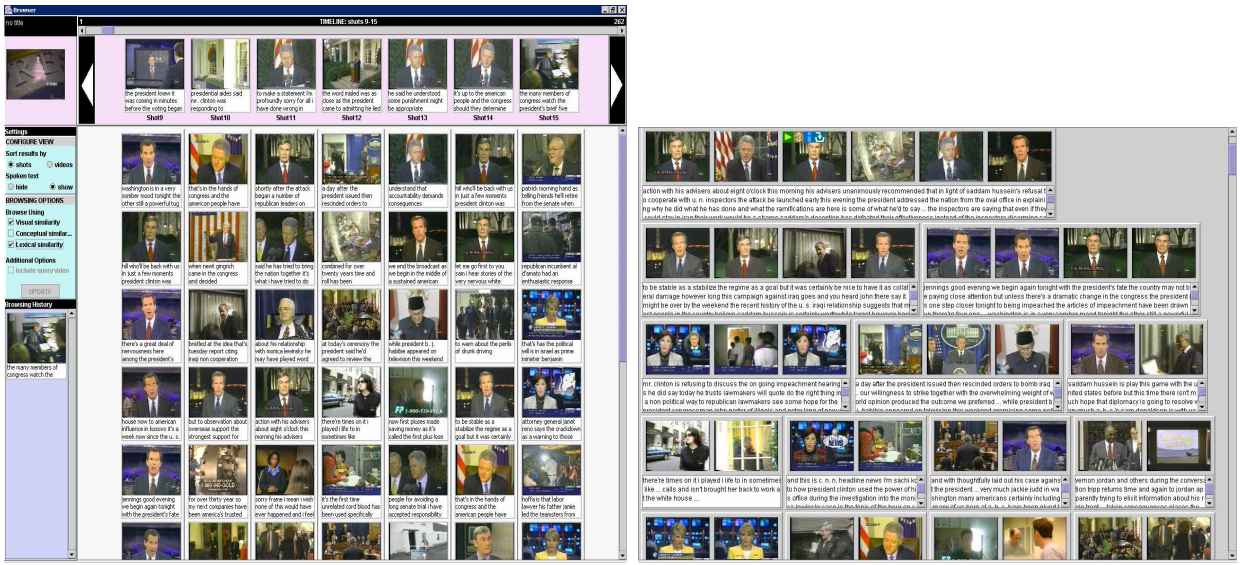
Interactive Browsing Interface

Our approach in the interactive search task relies on cluster-temporal browsing [9] and ways of viewing the result shots. The motivation is to reduce the effect of ambiguousness that is typically present in a traditional content-based example search. Currently, two approaches are dominant in typical video search and browsing systems. Systems either utilize a content-based search of video items or rely on tools that use more traditional time-line based organization of videos (fast forward, slide bars, hierarchical browsing). The disadvantage of the first approach is its incapability to associate computed features with the user's information need (ambiguity of content-based approaches). Second approach struggles with a problem of how to provide a holistic view over the linear temporal presentation (inefficiency of the time-line based browsing).

Cluster-temporal browsing combines both inter-video similarities and local temporal relations of video shots in a single interface. In interactive search, the role of the computer is to act as a 'humble servant', providing enough cues and dimensions for users to navigate through the vast search space towards the relevant objects. In VIRE, users can perform browsing that combines timeline presentation of videos with content-based retrieval. Cluster-temporal browsing implies that the video content is not utilized alone, but together with the temporal context of the video.

Figure 2 illustrates the browsing interface. In Fig. 2 (a) the topmost panel (first row of the key frame images) displays bounded sequence of temporally adjacent shots taken from a video file and presented as a chronological time-line. At any time, user can scroll through the entire video file to get a fast overview of the broadcast's key frames. The panel below the video timeline gives user another view of shot key frames, but this time they are from other videos in the database. This panel shows the results of multiple content-based queries created from the example shots at the top row. The query results are organized into columns and are shown in parallel order to create a similarity matrix. The columns show the most similar matches organized in top-down rank-order. Therefore the most similar shots are organized at the top of the matrix. This similarity matrix provides a view of the other similar videos in the database. Therefore a searcher who is browsing through the timeline of a news video file can instantly see large numbers of additional shots that have content similar to the query video sequence at the top. The similarity criterion is defined by the user-selected features. The engines that are available for browsing are visual and text search engines. User can select any combination depending on what conditions he/she wants to browse the database with.

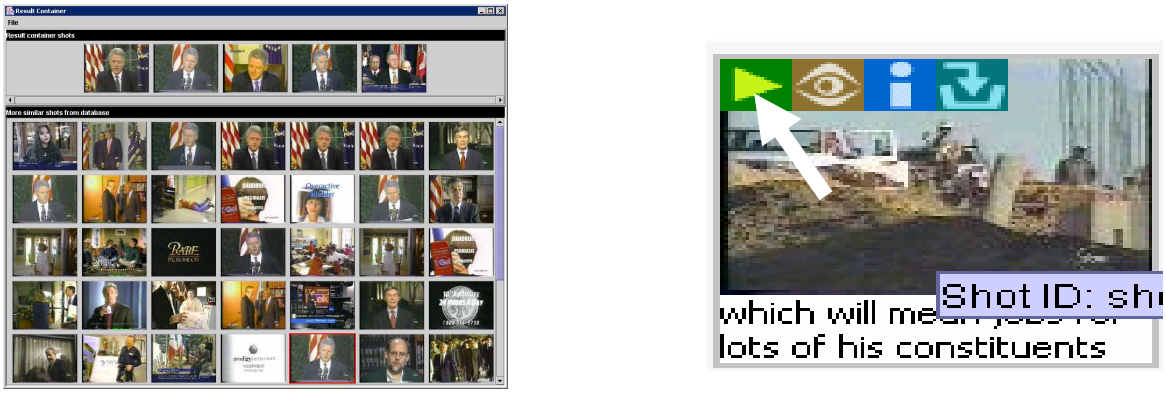
Navigation with the browser is straightforward. When user locates interesting shot in the similarity view, he can replace the current broadcast video (on top row) with the source of the interesting shot so that the shot of interest is located at the top of the middle column. After the video shots have changed at the top row, system re-computes the similarity view from the new set of shots. At any time, user can update the current view by changing to other feature combinations. Each transition caused by browsing the timeline in the current video brings new shots to the view. Because of the new content on the screen, the similarity view updates itself immediately.



(a) (b)

Figure 2 (a) Cluster-temporal browsing interface. The similarity view organizes result shots column-wise (b) Another organization of the similarity view: similar shots are grouped together if they originate from the same broadcast. The results are displayed using shot key frames. Also ASR text of the shot is visible under the key frame.

User can also select two ways to organize similar shots in a similarity view. The default view puts the results of a single search into columns under the top-row shots as show in Fig. 2(a). Another view groups the result shots based on their originating broadcast file. The largest group of results that are from the same video file are ranked highest. The shots are organized into rows and ranked to the screen from left to right and from top to down. Each shot key frame shows also the spoken content as text under the key frames. The requirements to update the similarity view are heavy, since the browsing speed should be close to real-time. To update a view, system must perform parallel query processing for several example-based queries. Multi-threaded index queries with efficient query cache provide reasonable access times even for the most complex feature configurations. Caching of results is essential when comparing different feature configurations having varying complexities.



(a) (b)

Figure 3. (a) Result container stores all the relevant selected shots and suggests more similar shots based on the selected ones. (b) fast action buttons are superimposed on each video item when mouse pointer is dragged over them.

New browser functions. The browsing interface for TRECVID 2004 experiments has been enhanced from the 2003 version by adding new elements for interactivity. First addition is the browsing history. Based on the questionnaires of the last year's user experiments, an option to reverse the navigational steps was often desired. Fig. 2(a) shows the history panel at the lower left corner of the interface. It collects the sequence of latest shots that were selected for browsing. Fig. 2(b) shows the grouping of results by videos, which is now optimised to show more results per screen. Figure 3(a) shows the result container. When the user finds a shot that is relevant, he adds it to the result container, which keeps a list of selected shots. Based on these shots, system generates new content-

based queries and suggests more similar clips from the database below the relevant shots. Figure 3(b) shows quick buttons that are superimposed on the key frames when mouse is dragged over them. These buttons can be used to fast playback the shot, browse the original broadcast file, display more information about the shot and add the shot to the result container.

2.2 TRECVID 2004 Experiments

MediaTeam submitted 10 result runs, six for interactive and four for manual search tasks. NIST provided 24 search topics that were used in the experiments. A topic contained one or more example clips of video or images and textual topic description to aid the search process.

NIST provided segmentation for the test video database (created by CLIPS-IMAG), from which 33367 video shots belonged to the feature search test collection. In addition to the main segmentation, 48818 sub-shots were given as a more accurate segmentation alternative for feature computation. However, since the TRECVID 2004 evaluation used the main segmentation, sub-shot segments were not allowed in the result submission. The results of collaborative annotation for TRECVID 2003, which resulted in annotation of feature search development collection, were allowed for system training also this year. Also the test collection from TRECVID 2003 was available for the system development and training. TRECVID 2004 test data was about 70 hours of ABC and CNN news from the year 1998.

Experimental Setup at MediaTeam

We configured our system to use master shot segmentation for the interactive search. Only visual and text search engines were used with equal weights, semantic search engine was disabled since the semantic features were not available at the time of our experiments. Concept search was enabled in manual search experiments, however. In manual experiments the search server was configured to use sub-shot segmentation inside the visual search engine. The sub-shot results of the visual engine were converted to the official shot segmentation so that the fusion of engines created valid TRECVID output. The visual similarity search utilised TCC and TGC features as described in 2.1. All provided media examples of the search topics were used in every visual search configuration throughout the manual experiments. In the feature fusion, search engines are being weighted with values from our previous experiments with TRECVID 2003 test collection and truth data [10]. Text search baseline run in manual experiments was based on only the original ASR transcripts from the NIST and the baseline text queries were constructed of the keywords in the original topic definitions text. Text query expansions were manually selected additional query words that elaborated search topic in the context of news videos. The expanded text queries contained approximately twice the search words of the baseline queries.

In manual queries a test user processed all 24 topics. VIRE system's manual search interface was used to generate four different configurational settings: [features] (weights text, semantic, visual) (run ID)

- [Original ASR + topic text keywords] (weights 1,0,0) (run M7)
- [ASR + Closed Captions + expanded query words] (weights 1,0,0) (run M8)
- [ASR + Closed Captions + expanded query words] & [visual features] (weights 4,0,2) (run M9)
- [ASR + Closed Captions + expanded query words] & [semantic features] & [visual features] (weights 4,1,2) (run M10)

The interactive search task was carried out by a group of 12 test users, from which four users had prior experience on searching with the system. Novice test users were mainly information engineering undergraduate students, having good skills in using computers, but little experience in searching video databases. Experienced users had used VIRE search system over 2 hours before arriving to test. Only one of the test people was a native English speaker. All of the test users were used to web searching. 12 users, 24 topics and two variants of VIRE system were divided into following configurations

- I1T: Variant A: S1[125-130],S7[131-136],S2[137-142],S8[143-148]
- I2VT: Variant B: S2[125-130],S8[131-136],S1[137-142],S7[143-148]
- I3T : Variant A: S3[125-130],S5[131-136],S6[137-142],S4[143-148]
- I4VT: Variant B: S4[125-130],S6[131-136],S5[137-142],S3[143-148]
- I5T: Variant A: S10[125-130],S9[131-136],S11[137-142],S12[143-148]
- I6VT: Variant B: S9[125-130],S10[131-136],S12[137-142],S11[143-148]

System variant A disabled the visual search engine so that searching was based entirely on ASR text search. System variant B enabled visual search engine together with text search. Each user did first six topics with one system configuration and then another six topics using another configuration. By setting the experiment into this configuration, the effect of learning was reduced between the system variants. The effect of fatigue was minimized with break and refreshments between system configuration change. The effect of learning within the topic sets was not controlled, most of the users processed the topics in numerical order. All users were given half an hour introduction to the system, with emphasis on search and browsing interface functions demonstrated with a couple of example searches. Users were told to use twelve minutes for searching a topic, during which they selected shots that seemed to fit to the topic description. The result submissions of 1000 shots were created using selected results as examples to retrieve more shots for the result set. In system variant A, additional shots were retrieved using text search engine only whereas variant B used a combined text and visual search with equal weights. The additional search time was added to the original time. Total duration of the experiment was about three hours. Users were told to fill a questionnaire about their experiences and they also filled the web questionnaires from TRECVID 2004 organizers. The machines that the system was running on were 0,8-2GHz PCs with Windows 2000/XP operating system installed. During the change of system configurations (halfway of the experiments), users were given refreshments and a break.

2.3 Search Results from TRECVID 2004

Mean average precisions for the 10 different search runs are shown in Table 1. The descriptions of the runs are found from Chapter 2.2. MAP shows the mean value of the average precisions in 24 search topics. The average time for interactive search was 12.18 with system variant A (text) and 13.27 with variant B (visual + text). For the system variant A, the average number of hits at depth 30 was 10.94 whereas for the variant B respective value was 10.47. The same values for hits at depth 10 were 6.14 and 5.72 respectively. Based on these numbers, the retrieval performance is on par with both system configurations. However, with system variant A search results are obtained approximately one minute faster on average.

With best manual configuration (M9) the average hits per topic at depths 10 and 30 were 2.08 and 4.88. From the total relevant shots returned one can note that in the manual experiments closed captions enhancement and manual text query expansion increase the amount of total relevant shots returned significantly.

Search Run ID	MAP	Total Relevant Shots Returned
I1T (interactive, novice users)	0.210	726
I2VT (interactiv, novice users)	0.179	678
I3T (interactive, experienced users)	0.212	767
I4VT (interactive, experienced users)	0.212	776
I5T (interactive, novice users)	0.212	723
I6VT (interactive, novice users)	0.201	721
Median (interactive)	0.181	497
Max (interactive)	0.337	980
OUMT_M7 (baseline)	0.072	535
OUMT_M8 (enhanced txt)	0.071	751
OUMT_M9 (enhanced txt+vis) (4,0,2)	0.077	767
OUMT_M10 (enhanced txt+sem+vis) (4,1,2)	0.072	782
Median (manual)	0.070	545
Max (manual)	0.119	782

Table 1. Results for search runs in 24 search topics

Some examples of the most successful topics for the interactive test users were 130 (hockey rink), 133 (Saddam Hussein) 134 (Boris Yeltsin), 135 (Sam Donaldson) and 136 (person hitting golfball).

3 Conclusions

Cluster-temporal browser gives many alternatives to pursue more meaningful results. With this system user can navigate between temporal contexts and content similarities in a single browsing interface. This gives the user better understanding about the relations between different video shots. The results indicate that the use of visual and text search with equal weights is as effective as using just text features in cluster-temporal browsing. These results do not account for the effects of semantic concept engine and weighting, both of which have been shown to increase the overall search performance in manual experiments [10].

Best manual configuration used visual and text search combined. The text search was weighted two times more than the visual search. Incorporated semantic concept search was degrading the search precision but surprisingly gave highest number of total relevant shots returned. The degrading effect in search precision is most likely caused by the problems with donated concept lists (see Section 2.1 for detailed description). According the run results, largest single improvement on recall was achieved by expanding the original topic text query with relevant keywords in the context of broadcast news.

Acknowledgments

The financial support of the National Technology Agency of Finland and the Academy of Finland is gratefully acknowledged.

References

[1] Ojala T, Rautiainen M, Matinmikko E & Aittola M (2001) Semantic image retrieval with HSV correlograms. Proc. 12th Scandinavian Conference on Image Analysis, Bergen, Norway, pp.621-627.

[2] Rautiainen M & Doermann D (2002) Temporal color correlograms for video retrieval. Proc. 16th International Conference on Pattern Recognition, Quebec, Canada.

[3] MPEG-7 standard: ISO/IEC FDIS 15938-3 Information Technology - Multimedia Content Description Interface - Part 3: Visual.

[4] Porter M (1980) An Algorithm for Suffix Stripping Program. Program, 14(3):130—137.

[5] Gauvain JL, Lamel L & Adda G (2002) The LIMSI Broadcast News Transcription System. Speech Communication, 37(1-2):89-108.

[6] Rautiainen M, Seppänen T, Penttilä J & Peltola J (2003) Detecting semantic concepts from video using temporal gradients and audio classification. International Conference on Image and Video Retrieval, Urbana, IL.

[7] Ho T, Hull J & Srihari S (1994) Decision combination in multiple classifier systems. IEEE Transactions on Pattern Analysis and Machine Intelligence 16(1): 66–75.

[8] IBM VideoAnnEx Server Page MPEG-7 (31.10.2004) http://mp7.watson.ibm.com/VideoAnnEx/

[9] Rautiainen M, Ojala T & Seppänen T (2004) Cluster-temporal browsing of large news video databases. 2004 IEEE International Conference on Multimedia and Expo, Taipei, Taiwan.

[10] Rautiainen M, Ojala T & Seppänen T (2004) Analysing the performance of visual, concept and text features in content-based video retrieval. 6th ACM SIGMM International Workshop on Multimedia Information Retrieval, MIR 2004, New York, NY.

[11] Video Dense Information Grinding (VideoDIG) (31.10.2004) http://www.research.ibm.com/VideoDIG/

[12] Ratcliff J.W. and Metzener D. (1988) Pattern Matching: The Gestalt Approach. Dr. Dobb's Journal, page 46, July 1988.