This paper describes our recent work on content-based visual query with emphasis on automatic visual feature extraction, efficient feature indexing, and feature-assisted search and browsing. Efficient techniques for manipulating compressed videos are included as well. We will discuss several research prototypes, including VisualSeek, which is a Java-based WWW application supporting localized color and spatial similarity retrieval; CVEPS (Compressed Video Editing and Parsing System), which supports feature-based indexing and editing of video in the compressed domain; and a hierarchical news video browsing and indexing system.

INTRODUCTION

Efficient and effective methods for indexing, searching, and retrieving images and videos from large archives are critical techniques required in visual information systems (VIS) applications. Users need a more powerful method for searching images than just traditional text-based query (e.g., keywords). Manual creation of keywords is too time-consuming for many practical applications. Subjective descriptions based on users’ input will be neither consistent nor complete. Also, the vocabulary used in describing visual contents is usually domain specific.

There is a recent effort, called content-based visual query (CBVQ) (Niblack et al., 1993; Jain, 1992), aimed at effective solutions to the above problem. Before a discussion of the technical issues involved in the CBVQ, we first briefly discuss its relationships with other image indexing/retrieval approaches.

There are several possible ways of indexing and retrieving visual material. From users’ point of view, the more methods available, the higher the flexibility they can exercise and adapt to the specific information they are seeking. However, from the system designer’s point of view, different methods imply different cost and efficiency. It is important to achieve good overall system performance. The traditional keyword-based retrieval methods can be extended to more general semantic-level descriptions such as “a red car in front of a house” or “a person running on the beach.” This type of retrieval requires semantic information given by users in the indexing stage. The second type of query allows users to specify a complete image or an image region as the query key. Specific
images can be retrieved based on similarity with the input image itself or the image features derived from the input image. This is usually called a query by image example. The last type of query is feature-based image retrieval. Visual signal features are extracted in the indexing stage and compared in the search stage to find the "similar" images/videos. Typical features include texture, color, shape, object layout, motion, camera operations, face, logo, associated audio and speech, etc. Some features are for still images and others are for videos although, in general, all still image features are applicable to video as well. The formulation of input features to the search engine can be provided by user's raw data (e.g., drawing and sketch) or user's selection from system templates. The population of feature sets from each image or video in the database can be automatic or semi-automatic (i.e., with user assistance). Knowledge of the application domain also helps significantly in developing reliable automatic retrieval techniques.

In CBVQ, the term "content" refers to the structure and semantics of images and videos at various levels, ranging from pixel patterns, physical objects, spatial/temporal structures, to high-level semantics of the visual material. The content-based approach is not intended as a replacement for the keyword approach. Instead, it is considered as a complementary tool, particularly for applications which have large data collections and that require a fast search response. Provision of the content-based visual retrieval techniques also brings in new synergy between the text-based information and the visual information of the same material. Fusion of different information channels (text and visual in this case) has been used to achieve performance improvement in multimedia databases such as news archives (Srihari, 1995).

A content-based visual query system requires several key components, including:

- choice of effective visual features and their integration with textual indexes;
- visual feature extraction (automatic or semi-automatic) and object segmentation;
- design of effective discrimination measures;
- efficient indexing data structure for high-dimensional feature space;
- efficient user interface for query specification and visual browsing;
- association with domain knowledge and other data types;
- exploration of functionalities in the compressed domain; and
- evaluation criterion and methodologies.

The above broad list reflects the broad cross-disciplinary nature of this area. However, to achieve the goal of searching images/videos based on visual content, a very challenging issue is visual content analysis and
automatic extraction of "prominent" features. The definition of "prominence" may depend on the actual application requirements, but a general bottom-up approach is as follows: start with a rich set of automatically extracted low-level generic visual features, then derive high-level semantics by applying the domain knowledge provided by users or applications.

We focus on issues directly related to image/video processing and associated algorithms for efficient indexing, computation, and visualization for improving overall system performance. Several groups have reported promising work in this area (Niblack et al., 1993; Pentland et al., in press; Zhang et al., 1993; Dimitrova & Golshani, 1994; Yeo & Liu, 1995; Hampapur et al., 1995; Mehrotra & Gary, 1995; Stone & Li, 1996; Sawhney et al., 1995). We contrast our work with others by using a fully automatic process for localized feature extraction (e.g., local color and texture regions), compressed-domain feature extraction without full decoding of compressed images/videos, and efficient user interfaces for users to specify localized features and their spatial layout.

We will describe two example prototype systems of content-based visual query—VisualSEEK and CVEPS. VisualSEEK is a fully automated content-based image query system which allows users to search images by localized colors and their spatial layout. It includes a Java-based Web interface for interactive visual content specification, integrated visual/textual search, and performance evaluation. It is the first CBVQ system which supports local content specification and integrated query using visual feature and spatial layout. CVEPS (Compressed Video Editing and Parsing System) is a software prototype of a video indexing and manipulation system which supports automatic video segment decomposition, video indexing based on key frames or objects, and compressed video editing. It is the first software-based system supporting both indexing and manipulation of compressed video. The compressed domain approach provides great benefits in reducing computational complexity, storage space requirements, and improving picture quality. Implementation of a simple editing function—e.g., random cut and paste—can be improved in speed by about 100 times by using the compressed-domain approach.

VISUALSEEK

VisualSEEK is a new content-based image query system that provides for querying by both image region visual properties and spatial layout. VisualSEEK is a hybrid system in that it integrates feature-based image representation with spatial query methods. The integration relies on a recently proposed representation of color regions by color sets (Smith & Chang, 1996). Color sets provide for a convenient system of region extraction through back projection. In addition, a decomposition of the
quadratic form of the color set distance function consists of terms that are easily indexed. This allows for the efficient computation of color set distance and the indexing of color sets. As a result, unconstrained images are decomposed into near-symbolic images which lend themselves to efficient spatial query.

CONTENT-BASED IMAGE QUERY

A recurring problem in image database research is how to efficiently retrieve items from the database that optimize some function. In content-based image query applications, this function approximates the perceptual similarity between the user's query image and target images in the database. In general, when the database is large and the image features have many dimensions, the exhaustive search of the database is not computationally expedient. Furthermore, recent approaches toward content-based image query have neglected two important aspects of visual perception—spatial information and spatial relationships.

SPATIAL IMAGE QUERY

A significant aspect of discriminating among images depends on the spatial locations and relationships between objects or regions within the image. However, the problem of content-based image query is only exacerbated by introducing multiple image regions and spatial information into the query process. This is due to the combinatorial explosion resulting from comparison among multiples of regions or objects. On the other hand, by representing images symbolically, spatial query methods compare the spatial relationships of symbols. However, spatial queries do not consider the similarity of the "symbols" such as that based upon visual features of objects or regions.

JOINT CONTENT-BASED/SPATIAL IMAGE QUERY

In VisualSEEk, we propose a new system that provides both feature comparison and spatial query for unconstrained color images. To illustrate (see figure 1), each image is decomposed into regions which have feature properties such as color, texture, and shape and spatial properties such as location and spatial relationships. The most desirable image query system allows users to query by both visual features and spatial properties. Recent approaches for image retrieval do not provide for both types of querying. The QBIC system (Niblack et al., 1993) provides querying of manually segmented regions by color, texture, and shape but not by spatial relationships. The Virage system (Bach et al., 1996) allows querying of only an image's global features such as color, composition, texture, and structure.
To solve the problem of integrating content-based and spatial image query, we decompose the parameters of the image distance function into two classes—intrinsic and derived variables. Then we design the representations for intrinsic variables such as region color, spatial location, and size to require minimal computation in matching. For example, color matching is achieved efficiently through color sets. Furthermore, the intrinsic variables are indexed directly to allow for maximum efficiency in queries. In this way, a query specified by the user is translated into pruning operations on intrinsic variables. The derived variables, such as region-relative locations and special spatial relations, are resolved only in the final stage of the query. This is because these evaluations have the highest complexity. The pruning performed by the queries on the intrinsic variables reduces the number of candidate images that need to be evaluated at the final stage.

UNIQUE FEATURES OF VISUALSEEK

VisualSEEK has joint image feature/spatial querying; automated region extraction; and direct indexing of color features. The VisualSEEK project has also emphasized several unique objectives in order to enhance the functionality and usability of image retrieval systems: (1) automated extraction of localized regions and features (Smith & Chang, 1996a), (2) querying by both feature and spatial information (Smith & Chang, 1996b), (3) extraction from compressed data (Chang, 1993), (4) development of techniques for fast indexing and retrieval, and (5) development of highly functional user tools. The VisualSEEK client application was developed in the Java language to allow for maximum functionality, client platform independence, and accessibility on the World Wide Web. As illustrated in Figure 2, VisualSEEK consists of several components: the set of user tools, the query server, the image and video retrieval server, the image and video archive, the meta-data database, and the index files. Currently, the VisualSEEK system allows searching on a test bed of 12,000
miscellaneous color images. The users can search for images using color and spatial attributes.

Figure 2. VisualSeek system architecture.

Figure 3. VisualSEEK user interface provides tools for sketching query regions, assigning region properties and positioning regions to form joint content-based/spatial image queries.
QUERY FORMULATION

The joint color/spatial queries are formulated graphically by using the VisualSEEK user tools as illustrated in figure 3. The user sketches regions, positions them on the query grid, and assigns them properties of color, size, and absolute location. The user may also assign boundaries for location and size. The relationships between regions are diagrammed by connecting regions. In this way, the interface provides for queries that include region features and combinations of both absolute and relative placement of the regions.

VISUALSEEK QUERIES

We present some example joint content-based/spatial queries in VisualSEEK. In the first example (see Figure 4[a]), the query (top) specifies the absolute location of a single region. The retrieved image (bottom) has the best match in features (color and size) to the query region and falls within the "zero distance" boundary diagrammed in the query. In the next example (see Figure 4[b]), the query specifies multiple regions. The retrieved image provides the best match in terms of the features and absolute spatial locations of the query regions. In the next example (see Figure 4[c]), the query specifies the spatial relationships of regions. The retrieved image has three regions that best match the features of the query regions and their spatial relationship satisfies that specified in the query. Finally (see Figure 4[d]), the query specifies both absolute and relative locations of regions. In this query, the match to the region positioned by absolute location (top left region in the query image) considers both the features and location of this region. The match to the other regions (the bottom two regions in the query image) at first

![Figure 4](image)

Figure 4. Example VisualSEEK queries (a) single region with absolute location, (b) two regions with absolute locations, (c) multiple regions with relative locations, (d) multiple regions with both absolute and relative locations.
considers only the features of these regions. In the last stage of the query, the spatial relationship of the regions is evaluated to determine the match.

The discrimination of images is only partly provided by features such as color, texture, and shape. Another important component is based upon the spatial locations and relationships of objects and regions within the images. The VisualSEEk system provides for the retrieval of images by image features and by the spatial locations, relationships, and sizes of color regions. In VisualSEEk, the image feature computation is efficiently integrated into the spatial querying methods. In the future, work will extend the VisualSEEk system to support other image features, such as texture and shape, in the task of joint feature-based/spatial image query.

CVEPS

CVEPS is a functional software-based prototype system for compressed video editing, indexing, and browsing. It provides automatic tools for video segment (i.e., shot) decomposition, special effect detection (e.g., zoom, panning, dissolve), moving objects extraction, key frame/object indexing, and nonlinear editing, mostly in the compressed domain without full decoding of the original compressed streams. By using the compressed-domain approach, real-time performance is achieved even with software implementation.

The compressed-domain approach offers many great benefits. First, implementation of the same manipulation algorithms in the compressed domain will be much cheaper than that in the uncompressed domain because the data rate is highly reduced in the compressed domain (e.g., a typical 20:1 to 50:1 compression ratio for MPEG). Second, given most existing images and videos stored in the compressed form, the specific manipulation algorithms can be applied to the compressed streams without full decoding of the compressed images/videos. Third, because full decoding and re-encoding of video is not necessary, we can avoid the extra quality degradation that usually occurs in the re-encoding process. We have shown earlier that for MPEG compressed video editing, the speed performance can be improved by more than 100 times, and the video quality can be improved by about 3-4 dB if we use the compressed-domain approach rather than the traditional decode-edit-encode approach (Meng & Chang, 1996).

The primary compression standard used in CVEPS is MPEG (MPEG-1 or MPEG-2). But the underlying approach and techniques are general enough to be applied to other video compression standards using transform coding and/or interframe motion compensation.

In order to allow users to manipulate compressed video efficiently, two types of functionalities are required: (1) key content browsing and search, and (2) compressed video editing. The former allows users to
efficiently browse through, or search for, key content of the video without decoding and viewing the entire video stream. The key content refers to the key frames in video sequences, prominent video objects and their associated visual features (motion, shape, color, and trajectory), or special reconstructed video models for representing video content in a video scene. The second type of functionalities, video editing, allows users to manipulate the object of interest in the video stream without full decoding. One example is to cut and paste any arbitrary segments from existing video streams and produce a new video stream which conforms to the valid compression format. Other examples include special visual effects typically used in video production studios.

SYSTEM COMPONENTS

CVEPS consists of three major modules: parsing, visualization, and authoring (see Figure 5). In the parsing module, MPEG-compressed video is first broken into shot segments. Within each shot, camera operation parameters (e.g., zooming, panning) are estimated. Then moving objects are detected and their shape and trajectory features are extracted.

![Figure 5. CVEPS System Overview.](image-url)
In the visualization module, the scene cut list and the camera zoom/pan information are used to extract key frames for representing each video shot. For example, a zoom-in video shot may be represented with the first frame of the shot and a detailed zoom-in frame or a video mosaic.

The key frames can be browsed with the hierarchical video scene browser based on scene clustering described in the next subsection (Zhong et al., 1996). Our content-based image query system, VisualSEEK, can be used to index and retrieve key frames or video objects based on their visual features and spatial layout. In the authoring module, we provide tools for cutting/pasting of arbitrary MPEG video segments and adding special effects such as dissolve, keying, masking, and motion effects (variable speed, stroke motion, etc.).

**Scene Clustering**

We have developed scene clustering techniques to help users to browse and search efficiently important content in a video sequence. Each scene can be assumed to have consistent video content in most cases, except special situations like fast changing foreground and fast camera motion. Therefore, at the scene level, visual characteristics can be explored to classify various video scenes and derive higher-level semantics. Given a large collection of video sequences, a hierarchical scene-browsing interface will be useful for users to quickly browse through the content contained in a long video sequence or multiple video sequences.

The hierarchical scene browsing system (Zhong et al., 1996) takes the list of detected scenes from CVEPS and organizes them into multiple levels according to three different criteria: (1) temporal order, (2) story (e.g., news story), and (3) visual features. Organization based on the temporal order groups scenes into multi-level clusters according to their sequential order in time. $N$ consecutive scenes form a basic segment in the lowest level, $M$ segments form a class in the next level, and so on until all scenes are included. This hierarchy ensures a fully balanced organization with the same number of scenes in each class.

The second browsing mode uses the knowledge of story boundary and groups all video scenes contained in the same story into a class. This allows users to quickly view different stories in a long video program without going through the low-level scenes. This is particularly useful in situations where users need to quickly find a particular story from a large video archive to meet short deadlines.

The third browsing mode organizes scenes based on their visual features. Users may want to find related scenes from different programs (e.g., related news of different days). Scenes with similar visual features (e.g., color, motion, and faces) may be assigned to the same class if their visual similarity is sufficient. One useful example is to find all anchorperson scenes automatically. This allows automatic detection of the news story...
boundary without user intervention. Currently, our system has implemented scene clustering using primitive features—i.e., by color and motion. Advanced features will be incorporated later.

REMOTE ONLINE EDITING

The client-server model used in CVEPS provides for economic design of the client's editing terminal. Users may use a lightweight viewing station to connect to a powerful CVEPS server. The emerging WWW technology also provides a very powerful platform for remote online editing. With a Java-enabled Web browser, the user can first browse through the keyframes in a video stream; query and retrieve desired video clips; and preview and edit video at a low resolution. Upon completion, the client software will generate a standard Edit Decision List (EDL). The EDL is sent to the server for generating a full resolution video. The CVEPS server will handle most of the computing-intensive tasks, such as stream parsing, and any required coding process. Once the actual manipulation and rendering of desired special effects is finished, users can request retrieval of the full-resolution video for stream download or real-time display. Note that multiple EDLs can be stored for the same video sequences before the final rendering. Final video output should be produced from as original a video source as possible to minimize multiple-generation quality loss.

Figure 6. The Compressed Video Editing and Parsing System User Interface.
CVEPS PROTOTYPE

We have developed a prototype of the CVEPS with a C/Unix/Motif graphic user interface (see figure 6). Every time an MPEG video is opened, its corresponding video buffer status is plotted against a timeline. This allows us to monitor and verify the bit rate of any video stream produced in CVEPS. The video bit rate needs to conform to some specification in MPEG and the parameters set in the encoded stream to avoid decoder abnormality (e.g., decoder buffer overflow or underflow).

The user may run the scene cut detection to extract a list of keyframes representing video shots. At any time, the user may invoke the MPEG software viewer and the interactive VCR control panel to do random search, step forward, fast forward/reverse, etc. The user can also use the mouse to highlight the time-line to select arbitrary video segments and apply “copy, cut, and paste” operations. The CVEPS also provides options of inserting special effects—such as dissolve, fade in/out, and wipe—between the connected segments.

CONCLUSION

We have discussed innovative and efficient techniques for image/video indexing, retrieval, and manipulation in large visual information management systems. We presented two working systems, VisualSEEk and CVEPS, to illustrate our unique research approach in content-based visual query and manipulation.

VisualSEEk is a new content-based image query system that provides for querying by both image region visual properties and spatial layout. Unconstrained images are decomposed to local regions with prominent visual features (color, texture) by automatic tools. Spatial relationships are indexed with direct data structures to support efficient spatial queries.

CVEPS demonstrates advanced video indexing and manipulation functions in the compressed domain. The video parsing tools support automatic extraction of key visual features—e.g., scene cuts, transitional effects, camera operations (zoom/pan), shape, and trajectories of prominent moving objects. These visual features are used for efficient video indexing, retrieval, and browsing. The editing tools allow users to perform useful video composing functions and special visual effects typically seen in video production studios. We contrast our compressed-domain approach with traditional decode-process-reencode approach with a quantitative and/or qualitative performance comparison.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under a CAREER award (IRI-9501266), by IBM under a Research
Chang, Smith, & Meng

Partnership (Faculty Development) Award, and by sponsors of the AD-VENT project of Columbia University. The scene clustering and hierarchical video browsing system includes joint work with Di Zhong and Horng Jiang Zhang.

REFERENCES


