A Novel Approach to Fine-grained Content-based Access Control for Video Databases

Nguyen Anh Thy Tran
FCGV Company
HoChiMinh City, Vietnam
t1tran@fcg.com

Tran Khanh Dang
Faculty of CSE
HCMC University of Technology
HoChiMinh City , Vietnam
dtkhanh@hcmut.edu.vn

ABSTRACT

The growing amount of multimedia data available to the average user has reached a critical phase where methods for indexing, searching, and efficient retrieval are expressly needed to manage the information load. In this paper, we present a hybrid video database model which is a combination of the hierarchical video database model and annotations using. In particular, we extend the original hierarchical indexing mechanism to add frames and salient objects as the lowest granularity level in the video tree with the aim to support multi-level access control. In conjunction, with the use of annotations, we give users more solutions to query for their interesting videos based on the video contents (content-based retrieval). In addition, we also tailor the original database access control model to fit the characteristics of video data. Our updated model supports both multiple access control policies, means a user may be affected by multiple polices, and multilevel access control, means an authorization may be specified at any video levels.

1 INTRODUCTION

The field of multimedia systems has experienced an extraordinary growth during the last decade. Consequently, the design and implementation of video database systems has become a major topic of interest.

With the huge amount of video information stored in archives worldwide, video databases have been researched for many years to introduce some efficient ways to manage this kind of data. Below are some criteria that a video database should satisfy:

- The first thing that should be satisfied is how to organize raw video data efficiently. It means the videos should be normalized to a standard form and compressed before being stored.
- Secondly, the video database access control scheme should be integrated with the database indexing structure in order that video database access control can be achieved more effectively.
- Thirdly, the flexibility and efficiency of transmitting video data through networks is an important consideration because most video databases are deployed over networks.
- Finally, control over the security of a video database system is important. Videos can be illegally accessed while being transferred over the network, or accessed directly into the database.

To achieve the above requirements, this paper proposes a video database system that supports content-based retrieval and multilevel access control.

2 VIDEO DATABASE MODELS

When very large video data sets come into view, video database models and indexing can no longer be ignored if one wants to support effective video retrieval and access control. In this section, a hierarchical indexing technique is introduced and extended to support multi-level access control.

A Hierarchical video database model

In order to control access efficiently, most video databases are designed as hierarchical structures such as the *semantic cluster tree* [1]. Within this structure, video contents are first partitioned into a set of semantic clusters; each semantic cluster is then partitioned into a set of sub-clusters and each sub-cluster may consist of a set of sub-regions. Using this indexing method, the system can handle multi-level access control efficiently.

A new proposed hierarchy video database with more granularity levels

The above model has many advantages but it also has some limitations. Firstly, the only video unit supported is video shot while users often interested in the whole video contains a certain shots. Secondly, the hierarchy tree is inflexible because in the case of extremely large databases, the tree level cannot be increased. Thirdly, this model cannot support access control at a frame and salient object granularity level. Finally, it loses most of the information needed for flexible content-based retrieval. Even though clusters are high semantic level extracted from other information, we still need to remain those information such as captions, audios, images etc. Given the above reasons, this paper suggests a new model as illustrated by figure 2.1 to tackle these issues.

To address the first restriction, two new video levels are introduced; *video* and *scene*, meaning that a complete video may contain some scenes and a scene

contain some shots. With this enhancement, a video database administrator can specify authorizations at video (most often), scene and shot levels. This paper also proposes to modify the original hierarchy of the video tree to use video groups which consist of videos or other groups instead of clusters, sub-clusters and sub-regions. With this amendment, the path from root to the leaves can be controlled with greater flexibility where new groups can be introduced or existing groups removed.

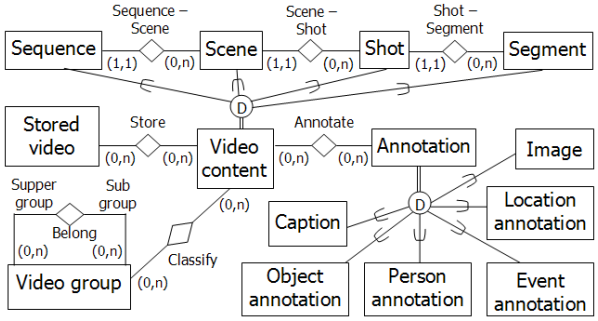

Figure 2.1 Extended video database

Along with the two above amendments, it is suggested that a new video element at the lowest granularity level called *video segment* be introduced. This element would prove very useful when applying access control to a frame or a salient object. Consider the example in Figure 2.2, where there are two users *A* and *B* within the system.

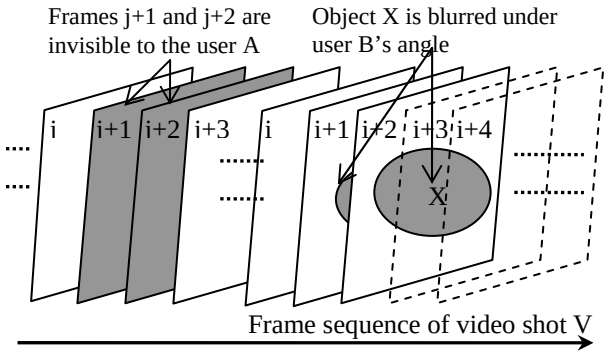

Figure 2.2 Segment example

A policy applies to user *A* that specifies that this user cannot view two frames $(j+1)^{th}$ and $(j+2)^{th}$ of video shot *V*. In addition, there is another policy that restricts user *B* from seeing object named *X* of video shot *V*. The easiest way to handle these two policies is to copy the whole video shot *V* to two appropriate versions. However, this solution will impinge on memory space since video data is often huge. Using segments is another solution which splits the video shot to certain segments and then copies the segments only when needed. In this example, the video shot *V* is split into 5 separate parts: (1) from the beginning to frame j^{th}, (2) frames *j+1* and *j+2*, (3) from frame $(j+3)^{th}$ to frame i^{th}, (4) frames *(i+1)* and *(i+2)*, (5) from frame $(i+3)^{th}$ to the end. With this solution, we only need to copy the segment 5^{th}, contains only two frames, into two versions *#1*, with original *X* object, and *#2*, with blurred *X* object. Then, when user A requires this video shot, the system will display the 1^{st}, 3^{rd}, 4^{th}- version #1 and 5^{th} segments while user B sees 1^{st}, 2^{nd}, 3^{rd}, 4^{th} – version #2 and 5^{th} segments.

The final adjustment is related to annotations. Since information in videos is quite "raw" and dispersed, it is almost impossible to achieve content-based access to videos unless some additional information is available. In order to enable flexible and intelligent access to videos, we somehow need to extract "keywords" which describe contents of videos. Typically "keywords" are useful for content-based access to videos include information on:

- what/who appears in the video,
- when the video was broadcast/recorded,
- where the video was recorded,
- what the video is about, etc

In order to achieve this goal, added annotations are required such as object annotation, person annotation, location annotation, event annotation, caption and image.

3 AN ACCESS CONTROL MODEL

In this paper, a content-based access control model is suggested which is reliant upon high level features extracted during the video processing stage. The goal of the system is to provide a flexible framework that can support different security levels against the video database.

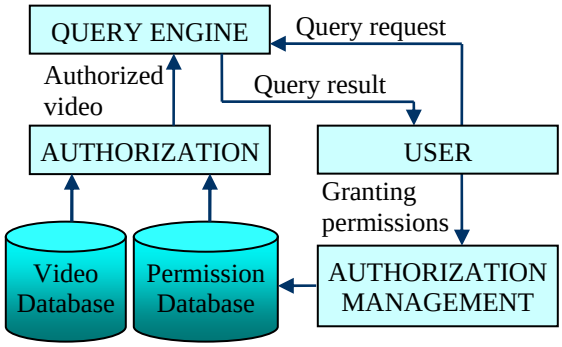

Figure 3.1 Video database access control architecture

The architecture of the system is illustrated in figure 3.1 with three main components. The *authorization* component is responsible for filtering authorized videos while the *query engine* searches for interesting videos that a user has requested. The final component, *authorization management*, handles granting permissions and ensures the consistency and integrity of the video database system.

3.1 THE AUTHORIZATION MODEL

In this section, an authorization model based on the flexible authorization model suggested by E.Bertino

and S. Jajodia in [1] is introduced. The model provides both multiple access control polices and a multi-level access control mechanism. This model also allows the administrator to specify multiple authorizations over any users or user groups (named *subject of the authorization*) against any video level such as videos, scenes or video shots.

Notations and definitions

This new model manages access control via authorizations. The subject of each authorization is a user or a user group. A group can contain some users and/or other user groups. The relationship between a subject s and a user group G_k can be either *direct* or *indirect* [1]. If s is a member of G_k, we count this relationship as direct, written $s \in_1 G_k$. In contrast, the relationship is indirect, written $s \in_n G_k$, $n > 1$, if there exists a sequence $<s_1, s_2, \ldots, s_{n+1}>$, such that $s_1 = s$, $s_{n+1} = G_k$ and $s_i \in_1 s_{i+1}$, $1 \le i \le n$. The sequence $<s_1, s_2, \ldots, s_{n+1}>$ is called a membership path of s to G_k, written $mp(s,G_k)$. Let $MP(s, G_k)$ represent a set of memberships of s to G_k, either direct or indirect.

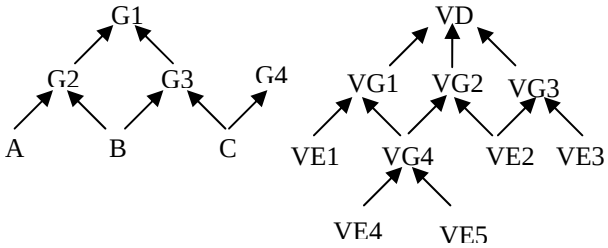

Figure 3.2 (a) user tree and (b) video tree

In a similar fashion to users, the video content of our system is also organized into an extended tree structure. Let V, VG and VO represent the videos, video groups and video objects (frames or salient objects) respectively. A video group can contain videos and other video groups. We use $v \in_k vg$ to denote the relationship where v belongs to vg with a relationship type that is direct ($k = 1$) or indirect ($k > 1$). We use $mp(v,vg)$ to represent the membership path of v to vg and $MP(v,vg)$ stands for the set of all paths of v to vg.

Authorization handles whether a user or user group can access (*positive authorization*) or cannot access (*negative authorization*) a video element. The authorization model within this paper is based upon the multi-level video access control model described in [14]. However, within this new system, the target of each authorization can be a node on the video content tree instead of a single table. Also supported are two kinds of authorization called *hard* (authorization that cannot be overridden) and *soft* (authorization that can be overridden).

Let U denote all users, G the set of user groups, $S = U \cup G$ the set of all subjects, V the set of video contents, VG the set of video groups, $VD = V \cup VG \cup VO$ the set of all video elements, AZ the set of all authorizations in our system. Authorizations can be defined as follows.

Definition 3.1 (Authorizations) An authorization is a 5-tube of the form (s, v, pt, g, at) where $s \in S$, $v \in VD$, $pt \in (+, -)$, $g \in U$, $at \in \{soft, hard\}$

The authorization states that s has been granted (if $pt =$ "+") or denied (if $pt =$ "-") access permission on the video element v by user g with authorization type is at (*soft* or *hard*). Given an authorization a, let $s(a)$, $v(a)$, $pt(a)$, $g(a)$, $at(a)$ denote the subject, target, access type, grantor and authorization type, respectively.

Since a user can belong to a number of different user groups, he or she can be affected by multiple authorizations and some of them have opposite access types over a video element. With this in mind we need to define rules to decide which authorization has priority. Our overriding authorization model is a user-driven one, meaning it prioritizes the authorizations based on the relationship between their subjects. The authorization that has a more detailed subject will have higher priority.

Definition 3.2 (Overriding authorization) Consider p_i and p_j which are two different authorizations, p_i overrides p_j over user s against video element ve, written $p_i >_{s,ve} p_j$, $s \in_m s(p_i)$, $s \in_n s(p_j)$, $m,n \ge 0$, $ve \in_l v(p_i)$, $ve \in_k v(p_j)$, $l, k \ge 0$, if any of the following conditions is satisfied:

- $at(p_i) > at(p_j)$, means $at(p_i) = hard$ and $at(p_j) = soft$
- $at(p_i) = at(p_j)$ and $s = s(p_i)$, $s != s(p_j)$
- $at(p_i) = at(p_j)$ and ($\forall mp \in MP(s, s(p_j))$: $s(p_i) \in mp$ or $\exists s' \in mp$, $\exists p' \in AZ$, $s' \neq s(p_j)$, $s' \notin_k s(p_i)$, $p' >_{s',ve} p_j$)

The above definition can be explained as the following:

- p_i overrides p_j if the authorization type of p_i is *hard* while p_j's authorization type is *soft*
- p_i overrides p_j if p_i and p_j have the same authorization type and p_i is applied over s directly while p_j is not
- p_i overrides p_j if p_i and p_j have the same authorization type and for all membership path mp of s to $s(p_j)$, either $s(p_i) \in_k mp$ or exists $s' \in mp$, $p' \in AZ$ and p' override p_j over user s against video element ve.

Definition 3.3 (Conflict) Two authorizations p_i and p_j are in conflict with respect subject s and video element v, written $p_i <>_{s,v} p_j$, with $s \in_m s(p_i)$, $s \in_n s(p_j)$; $v \in_l v(p_i)$, $v \in_k v(p_j)$; $i, j, l, k \ge 0$, iff $pt(p_i) \neq pt(p_j)$ and neither $p_i >_{s,v} p_j$ *nor* $p_j >_{s,v} p_i$.

Within the proposed system, conflicts are avoided by validating any actions that may cause the potential

conflict to occur. The detail of this validation will be described in section 3.3.

Authorization algorithm

To make sure users can only view video contents they are permitted to access, this paper suggests an algorithm to retrieve the appropriate videos based on the actor and the set of authorizations on the system.

Definition 3.4 (Video database projection) The projection of a video database *VD* with respect to a video element ve, written $\prod_{ve}(VD)$, is a set of video ve_i such that $ve_i \in_k ve$, $k \geq 0$. It means $\prod_{ve}(VD)$ only contains the child nodes of *ve* in the hierarchical video tree. In some places, *VD* is omitted and we use $\prod ve$ stands for $\prod_{ve}(VD)$.

$$\prod_{ve}(VD) = \{v: v \in VD, v \in_k ve, k \geq 0\} \quad (3.1)$$

Definition 3.5 (Video database prune) Consider a set of videos $VS = \{ve_1, ve_2, \ldots, ve_n\}$, the result after *VS* is pruned, written $\angle(VS)$, is a set that contains the elements of *VS* and each one is not a child of any other elements inside *VS*. It can be described formally as follow.

$$\angle(VS) = VS - \{v_i : v_i \in VS, \exists v_j \in VS, v_i \in_k v_j, k>0\} \quad (3.2)$$

Next is the algorithm to filter the list of video contents that a user can access. Firstly, it will get all video elements granted to the user by positive authorizations. Then, it collects the video elements that are inaccessible to that user. This list contains all video elements was granted by negative authorizations except the video contents that the negative authorization is overridden by a positive one.

Algorithm 3.1: Filter user accessible videos
INPUT: Id of the user, *u*
OUTPUT: *AV* (authorized video) – list of accessible videos
METHOD *authorizeVideo(u)*
initialize *AV* to be empty
let *pos_permission* and *neg_permission* are lists of positive and negative permissions respectively
let *UV* is a list of videos that the user cannot access
let *TV* is a temporary list of videos
for each permission $p \in P$ do
if ($u \in_k s(p)$) then
if ($pt(p) = +$) then add p to pos_permission
else add p to neg_permission
for each permission $p+ \in pos\_permission$ do
$AV = AV \cup v(p+)$
for each permission $p- \in neg\_permission$ do {
$uv = v(p-)$
for each permission $p+ \in pos\_permission$ do {
$TV = \angle(\prod v(p+) \cap \prod v(p-))$
for each video element $ve \in TV$ do
if ($p+ >_{u,ve} p-$) then $uv = uv - ve$
}
$UV = UV \cup uv$
}
$AV = \angle(AV - UV)$
return AV
END *authorizeVideo*

3.2 QUERY ENGINE (VIDEO RETRIEVAL)

This component collects requests from end users and searches through the authorized videos to retrieve those relevant and returns them to the users. The query engine must be reliable, meaning that users can only access those videos to which they have been granted permission. The component must also be flexible, in order to support various means for the users to reach their interesting videos. Bertino suggested a system to support two access methods named querying and browsing [1].

Querying mode

Under the querying mode access control, a user submits a query to obtain access to a video element. A query is a n-dimension tube $(x_1, x_2, \ldots, x_n)$ of which x_i, $i=1..$ n is a value of the feature i^{th}. Below is the algorithm to retrieve video elements based on the features input.

Algorithm 3.2 Querying access control mode
INPUT: Id of the user, *u*
A query with $(x_1, \ldots, x_n)$ format where x_i is a feature's value
OUTPUT: *AIV (authorized interesting video)* – set of authorized filter video elements
METHOD *queryVideo(u, $(x_1, \ldots, x_n)$)*
AV = authorizeVideo(u)
if (*AV* is empty) then *AIV = empty*
else *AIV = solve_query(request, AV)*
return *AIV*
END *queryVideo*

This access control procedure consists of two main steps: (1) Firstly, it narrows the search space by filter a list of videos the user can access; (2) Secondly, it calculates all matching ranks between each video element in the *AV* list and the input query. Then, the system returns the result which will be videos ordered by their similarities with the input query.

'Solve_query' takes $x = (x_1, x_2, \ldots, x_n)$ as an input and calculates the similarity between the *x* vector and each authorized video. Then, it only keeps N videos which have the highest similarity measures which exceed a predefined threshold.

Browsing mode

Under the browsing access control mode, a user browses and navigates through video groups without specify searching criteria. Browsing refers to a technique or a process where users skip through information rapidly and decide whether the content is relevant to their needs. Browsing video databases should be like scanning the table of contents and indices of a book to quickly get a rough sense of content and gradually focus on particular chapters or sections of interest.

3.3 AUTHORIZATION MANAGEMENT

The main purpose of this component is to maintain the consistency and integrity of the system. It is responsible for validating all actions that may cause unsolvable conflicts to occur.

With two types of authorization – *soft* and *hard*, we may have three kinds of relationship between the authorizations: *hard – hard*, *hard – soft* and *soft – soft*. The second relationship (*hard - soft*) cannot be a conflict because a hard authorization always overrides a soft one. In addition, to prevent the conflicts between hard authorizations, this newly proposed system would only accept negative hard authorization.

Finally, there is only the last relationship, *soft – soft*, needed to be verified for conflicts. Below are three actions that may cause a conflict to occur:

- Adding a new authorization
- Adding an existing user subject to a group
- Adding an existing video element to a group
- Deleting an existing authorization

For each kind of action, we suggest a different algorithm to check confliction individually.

Check confliction when adding a new authorization

When a new authorization *p(s, v, pt, g, at)* is added to the system, a conflict may occur over s' children nodes in the user tree. Consequently, the system must verify confliction between *p* and each authorization *p'* affects any children of *s*.

Check confliction when adding an existing user subject to a group

When adding an existing user or user group *s* to another user group *g*, *s* will inherit all authorizations affecting *g*. Thus, we need to check conflict between a set containing the authorizations affecting g, named *GP*, and another set *SP* containing the authorizations affecting *s* and its children.

Check confliction when adding an existing video element to a group

Assuming we are adding an existing video element *v* to a video group *vg*. The fact that two authorizations *p1* and *p2* can only conflict if they affect at least one common video, means we only verify confliction between the authorizations affecting *v* and all its child nodes in the video tree.

Check confliction when deleting an authorization

When an authorization *p* is deleted, subject s(p) and its children will be affected again by the authorizations p' which was overriden by p. Hence, we must verify confliction between all authorizations p' that currently affect to s(p) and the authorizations affect to s(p)'s children.

4 CONCLUSION AND FUTURE WORK

In this paper, an approach has been proposed to support both multiple access control policies and a multilevel access control mechanism. This technique combines video indexing mechanisms with a hierarchical organization of video contents, in order that different classes of users can access different video elements or even the same video element based on their permissions. This multi-level video access control mechanism approach employs a cluster-based indexing technique. It has also proposed certain common key features such as text, caption, audio etc to provide flexible query against the video database.

Future work includes comprehensive testing of the proposed models and the implementation of the materialization approach for performing access control.

REFERENCES

[1] E. Bernito, J. Fan, E. Ferrari, M-S. Hacid, A.K. Elmagarmid, X. Zhu: *A Hierarchical Access Control Model for Video Database Systems*, ACM TOIS, 21(2), 2003, 157-186.

[2] S. Deb: *Video Data Management and Information Retrieval*, IRM Press, 2005.

[3] I.E.G. Richardson: *H.264 and MPEG-4 Video Compression*, John Wiley & Sons, 2003.

[4] J-Y. Zhang: *Advances in Image and Video Segmentation*, IRM Press, 2006.

[5] B-L. Yeo, B. Liu: *Rapid Scene Analysis on Compressed Video*, IEEE Trans Circuits & Systems for Video Technology, 5(6), 1995, 533-544.

[6] H-J. Zhang, A. Kankanhalli, S. Smoliar, S. Tan: *Automatically Partitioning of Full-Motion Video*, Multimedia Systems, 1(1), 1993, 10-28.

[7] R. Hjelsvold, R. Midtstraum: *Modelling and Querying Video Data*, VLDB 1994, pp. 686-694.

[8] J. Calic, E. Izuierdo: *Efficient Key-Frame Extraction & Video Analysis*, Proc. Intl Conf on Information Technology: Coding & Computing, 2002.

[9] H. Kosch: *Distributed Multimedia Database Technologies Supported by MPEG-7 and MPEG-21*, CRC Press, 2003.

[10] B. Furht, O. Marques: *Handbook of Video Databases: Design and Applications*, Taylor & Francis Group, 2005.

[11] H.J. Zhang: *Content-based Video Browsing and Retrieval*, CRC Press, 1999.

[12] N. Adam, V. Atluri, E. Bertino, E. Ferrari: *A Content-based Authorization Model for Digital Libraries*, IEEE TKDE, 14(2), 2002, 296-315.

[13] A. Baraani-Dastjerdi, J. Pieprzyk, R. Safavi-Naini: *A Multi-level View Model for Secure Object-oriented Databases*. Data & Know. Eng, 23(2), 1997, 97-117.

[14] E. Bernito, S. Jajodia, P. Samarati: *Supporting Multiple Access Control Policies in Database Systems*, IEEE Symp on Security & Privacy, 1996, pp. 94-107.