Video Comparison Using Tree Matching Algorithms

Chung Wing Ng, Irwin King and Michael R. Lyu

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong SAR

Abstract *This paper proposes a video structure matching framework based on a tree matching algorithm which can be used as an effective video retrieval tool in a media management system. We define two types of specialized tree matching algorithms. The first is the non-ordered tree matching and the second is the ordered tree matching. The first algorithm is not constrained by the temporal ordering of the video, whereas the second algorithm takes the temporal ordering into account. Moreover, these two algorithms are special since they both match a well-structured tree-like video model having the same tree depth. Our experiments on a small set of various videos demonstrate that the proposed tree matching algorithms produce similar ranking results to what human will produce.*

Keywords: video comparison, video similarity, video structure, tree matching

1 Introduction

Content-based information retrieval is a very popular research topic in recent years. In particular, video content-based retrieval is gaining more attention because there is a large volume of video content available. Although many researchers have investigated video content retrieval problems [1, 2], there are still many interesting problems to be solved. One of such problems is the video matching problem [3, 4, 5]. This is still a challenging problem since it is difficult to extract video features that represent the content for matching, and there are many possible algorithms to match the temporal ordering of video features. Since a video can be successfully decomposed into a hierarchical tree structure [6], we propose two tree matching algorithms. The first one is the non-ordered tree matching algorithm and the other is the ordered tree matching algorithm. They are different because the ordered tree matching algorithm is constrained by the temporal ordering but the non-ordered tree matching is not.

This paper is organized as follows. In the next section, the formation of the tree structure for a video is introduced. In Section 3, two tree matching algorithms are proposed. Section 4 describes the evaluation of the proposed tree matching algorithms. Finally, conclusion and future work are in Section 5.

2 Tree Structure of Video

A video can be decomposed into a well-defined structure. The structure consists of 5 levels [1, 6].

- Video Shot - It is an unbroken sequence of frames recorded from a single camera. It is the building block of a video.
- Key Frame - It is the frame which can represent the salient content of a shot.
- Video Scene - It is defined as a collection of shots related in the video content and temporally adjacent. It depicts and conveys the concept or story of video.

- Group of Shots - It is an intermediate entity between the physical shots and video scenes. The shots in a video group are visually similar and temporally close to each others.

- Video - It contains all the components above.

The tree structure of a video is constructed in a bottom-up manner starting from the shot level. Different methods on detecting boundary of shots have been proposed [7, 8]. Most of them can be categorized into several types; however, the histogram-based method is one of the most popular and efficient approaches. By applying these boundary detection methods, a video is divided into shots, which consist of a sequence of similar frames. People suggest building up the group of shots by grouping similar shots together [1, 6]. In this way, the number of video fragments can be reduced, and grouped analysis would be more effective. At the video scene level, it is composed of content related groups. Rui proposed that we could construct a video scene by the temporal information defined. These video scenes constructed should be more understandable than video groups and video shots described above. According to Rui's method [6], a video can extract a hierarchical representation. The hierarchy is demonstrated in Figure 1. The hierarchy can then be transformed into a structured format as shown in Figure 2. This structure can be regarded as a specialized tree with its tree depth always equal to four and the root is at the Video level. Based on this tree structure of video, tree matching algorithm is proposed.

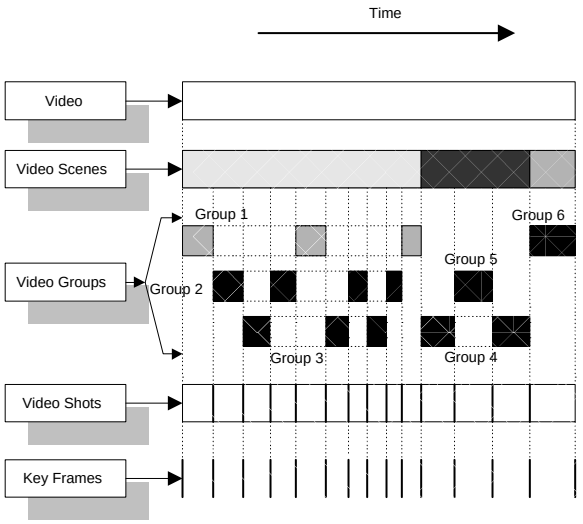

Figure 1: Hierarchical Representation of a Video

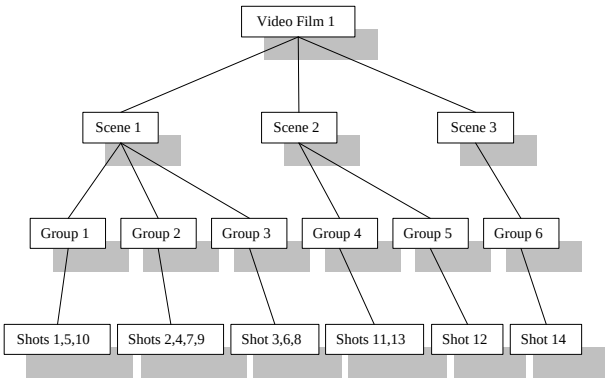

Figure 2: Structured Video

3 Tree Matching on Video

The tree matching algorithm for video is different from a general tree matching algorithm since the video tree we generated is well structured. The matching process starts from the top of the tree and proceeds to the next sublevel in an orderly manner, i.e., scene to scene, group to group, and shots to shots. Similarity measure is calculated at each corresponding level between the two video trees. We present two approaches to calculate the similarity of two video trees. One is the non-ordered tree matching algorithm and the other is the ordered tree matching algorithm. In the following sections, we detail these two heuristics based on two video features, i.e., color histogram and shot style.

The color histogram video feature is useful for matching the global color content of frames in the video. We can make use of the histogram difference between two frames to determine their visual similarity. If the difference is

small, the frames are similar; otherwise, these frames are different. In our algorithm, since a shot is a sequence of frames with similar content, a key frame which is the first frame in the sequence, is used to represent the whole shot. Then the color histogram of the key frame is used to compare with the color histogram from the key frame of another shot.

The shot style feature is composed of the camera motion and the length of the shot. The camera motion consists of zooming, horizontal movements, vertical movements, and still which means that there is no camera motion. In a shot, there could be many camera motion segments. For examples, a shot of a person may consist of zooming in and zooming out camera segments. In our system, we use the first camera motion segments to represent the camera motion for the shot. The length of the shot is the summation of all the camera motion segment durations in the shot. The camera movement and the length of the shot can reflect the pace of the video. For example, if a shot is short and the camera moves in different directions, we would expect that the video has a fast pace. The pace of video can help us to determine the type of video since we know that action videos are faster and artistic videos are typically slower.

3.1 Non-ordered Tree Matching

In the non-ordered tree matching, the shots are matched without any temporal sequence constraints. In other words, this method is able to match features in any order. The detail of the algorithm is described in the following paragraphs.

The algorithm examines the structural trees of two videos in a top-down manner, i.e., from the Video level to the Shot level. However the scoring of similarity of the video is propagated from bottom-up.

At Video, Scene and Group Levels

There are three steps to work out the feature similarities for the current level. First, the algorithm needs to retrieve the feature similarities of all children by traversing down the tree. For example, when we need to find the similarities between two videos, we need to know how similar their scenes are. When all the child feature similarities are tabulated, we calculate the feature similarities of the current level with a $MaxSum()$ function, in the second step. In the third step, we propagate the feature similarities of the current level to the parent.

The $MaxSum()$ function is used to sum up the similarities of the best match of the children and then return the normalized value of the sum. Figure 3 shows an example best match in the tabulated child similarities. Figure 4 demonstrates the matching in tree format. To calculate the sum, we can add up the maximum score at each row. However, the numbers of scenes, groups and shots in videos are not the same, the tabulated child feature similarities is not in square shape, and there are one to multiple mappings. For example, the fifth row in Figure 3, scene 5 from one video matches both scene 1 and 6 of another video. Then, the sum calculated is different if we take the summation of column maximum instead of row maximum. Therefore, in our algorithm, we set the feature similarity of the current level to be the mean value of the normalized column sum and row sum.

$$FeatureSimilarity = (\frac{maxcolsum}{numcol} + \frac{maxrowsum}{numrow})/2$$

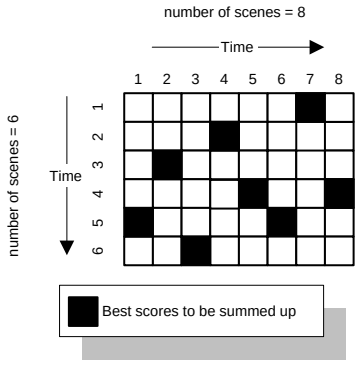

Figure 3: MaxSum()

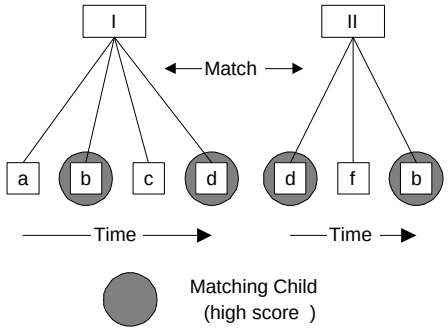

Figure 4: Non-ordered Tree Matching

At Shot Level

The algorithm calculates the feature similarities based on the shot features, which are the color histogram and the shot style.

The color histogram similarity is calculated using the key frames of shots. A distance function is used to compare the color histograms of the key frames [9]. The result that is the difference of the color histograms is normalized. Then the color histogram similarity is defined as follow.

$$ColorSimilarity = 1 - \text{Result of distance function}$$

The shot style feature similarity is set to be the ratio of shot durations when the representative camera motions are the same. For example, the camera motions of two shots are the same, and $duration_1$ is greater than $duration_2$, then the shot style feature similarity is equal to $duration_2$ divided by $duration_1$.

$$StyleSimilarity = \frac{duration_2}{duration_1}$$

After calculating both feature similarities, the algorithm propagates the shot level feature similarities to the upper level.

3.2 Ordered Tree Matching

The ordered tree matching algorithm is different from the non-ordered tree because it considers the temporal ordering of the shot features. It allows only matching of feature similarities with temporal constraints. Then the score of similarities propagated up is the summation of feature similarities for the best-ordered match.

An ordered tree matching is significant because it can capture the difference in video similarity due to the changes of features ordering. The reordering of features can form a different tree structure. However, the non-ordered algorithm cannot detect these kinds of structural differences. An ordered matching algorithm is designed to tighten the similarity measurement by the temporal sequences constraints, so that the features ordering are considered.

In this algorithm, we used a $MaxOrderedSum()$ function instead of the $MaxSum()$ function in non-ordered matching. The $MaxOrderedSum()$ function considers the ordering while finding out the sum of feature similarities for the best match. We use the dynamic programming technique to calculate the best score [10]. An example best match on tabulated child similarities is shown in Figure 5. The selected set matches a sequence of scenes from both videos. Figure 6 demonstrates this matching function in the tree format. There are four steps for our algorithm to find out the feature similarity of the current level.

Step 1: Initialize a matrix D with elements equal to zero.

Step 2: Fill in D according to the following rule.

$$D(i+1, j+1) = max(D(i,j) + ChildSim(i,j), D(i, j+1))$$

Step 3: When D is filled, the sum of child similarity for the optimal match is located at $D(numrow + 1, numcol + 1)$.

$$sum = D(numrow + 1, numcol + 1)$$

Step 4: Finally, we normalize the sum to be the feature similarity of the current level. The sum is divided by the number of rows and number of columns of the child similarities ta-

ble respectively. Then, we take the mean value to be the feature similarity of the current level.

$$FeatureSimilarity = (\frac{sum}{numrow} + \frac{sum}{numcol})/2$$

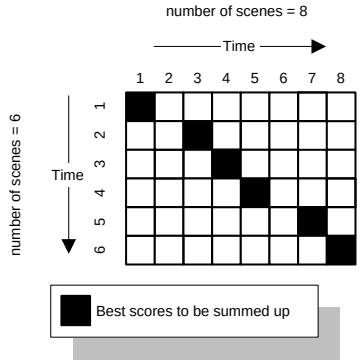

Figure 5: MaxOrderedSum()

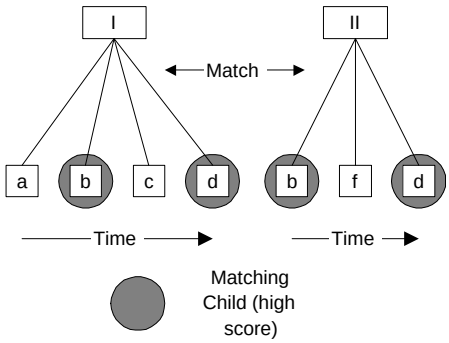

Figure 6: Ordered Tree Matching

4 Evaluation

In this section, the proposed tree matching algorithms will be evaluated by comparing the results of a small set of videos with the human's ranking results; one example is in Figure 7. Some information of the videos is shown in Table 1. The human's ranking results of the videos are shown in Table 2 and Table 3. There are 5 videos matching with each others using the proposed algorithms.

Table 1: Video Tree Structure Information

Videos	Number of shots	Number of groups	Number of scenes
Video 1	12	4	3
Video 2	14	5	2
Video 3	16	6	3
Video 4	18	6	2
Video 5	27	9	6

Table 2: Human's Ranking for Color Histogram Feature

Ranking Videos	Most Similar 1	2	3	Least Similar 4
Video 1 (V1)	V2	V3	V4	V5
Video 2 (V2)	V1	V4	V3	V5
Video 3 (V3)	V1	V2	V4	V5
Video 4 (V4)	V2	V1	V3	V5
Video 5 (V5)	V2	V1	V3	V4

4.1 Apply Non-ordered Tree Matching

According to the feature similarity scores calculated by the non-ordered tree matching algorithm, we rank the similarities between each video and the others. For example, when we match video 1 with the other 4 videos, if we find that video 2 have the highest similarity score, video 2 is the most similar one to video 1. The ranking results from non-ordered tree matching is shown on Table 4 and 5. We find that the results are quite similar to that of human. Hence, the algorithm can find out which

Table 3: Human's Ranking for Shot Style Feature

Ranking Videos	Most Similar 1	2	3	Least Similar 4
Video 1 (V1)	V2	V3	V4	V5
Video 2 (V2)	V1	V3	V4	V5
Video 3 (V3)	V1	V2	V4	V5
Video 4 (V4)	V2	V1	V3	V5
Video 5 (V5)	V2	V3	V1	V4

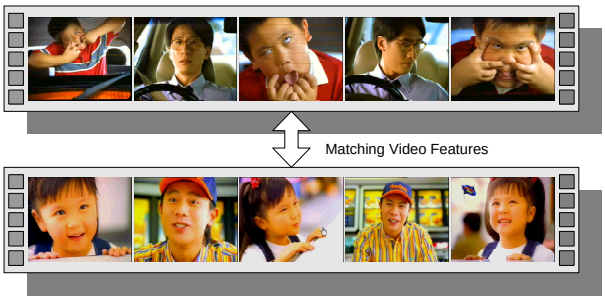

Figure 7: Matching Video Features

videos are more similar.

Table 4: Ranking Results of Non-ordered Tree Matching for Color Histogram Feature

Ranking Videos	Most Similar 1	2	3	Least Similar 4
Video 1 (V1)	V3	V2	V4	V5
Video 2 (V2)	V4	V1	V3	V5
Video 3 (V3)	V1	V2	V4	V5
Video 4 (V4)	V2	V1	V3	V5
Video 5 (V5)	V3	V2	V1	V4

Table 5: Ranking Results of Non-ordered Tree Matching for Shot Style Feature

Ranking Videos	Most Similar 1	2	3	Least Similar 4
Video 1 (V1)	V3	V2	V4	V5
Video 2 (V2)	V4	V3	V1	V5
Video 3 (V3)	V1	V2	V4	V5
Video 4 (V4)	V2	V3	V1	V5
Video 5 (V5)	V2	V3	V1	V4

4.2 Apply Ordered Tree Matching

Similar to the ranking in non-ordered tree matching, we rank the videos according to the result of the ordered tree matching. The ranks are shown in Table 6 and Table 7. The ranking results are also quite similar to that of human. That means the proposed algorithm can rank the similarities of the videos similar to human.

Table 6: Ranking Results of Ordered Tree Matching for Color Histogram Feature

Ranking Videos	Most Similar 1	2	3	Least Similar 4
Video 1 (V1)	V3	V2	V4	V5
Video 2 (V2)	V4	V5	V2	V3
Video 3 (V3)	V1	V4	V2	V5
Video 4 (V4)	V2	V3	V1	V5
Video 5 (V5)	V2	V1	V3	V4

Table 7: Ranking Results of Ordered Tree Matching for Shot Style Feature

Ranking Videos	Most Similar 1	2	3	Least Similar 4
Video 1 (V1)	V3	V4	V2	V5
Video 2 (V2)	V4	V5	V3	V1
Video 3 (V3)	V4	V1	V2	V5
Video 4 (V4)	V2	V3	V1	V5
Video 5 (V5)	V2	V4	V3	V1

5 Conclusion

There are two tree matching algorithms for video proposed in this paper. They are the non-ordered tree matching algorithm and the ordered tree matching algorithm. They score the similarity of videos using two video features extracted at the shot level. These features are the color histogram feature and the shot style feature. Evaluations have been done on comparing the video rankings of our algorithms and human, we found that the results are quite similar. Therefore we can conclude that the proposed algorithm is effective for matching the feature similarities of videos.

Our algorithms can be adapted to match video features other than the color histogram and shot style. Textual information and pattern of video sequence are the example features, which can be extracted from a video. Then we can make use of these new video features to improve the matching accuracy of our algorithms.

Acknowledgements

The work described in this paper was fully supported by the Research Grants Council of the Hong Kong Special Administrative Region, with Project Numbers CUHK4193/00E and CUHK4407/99E.

References

[1] J.L. Koh, C.S. Lee, and A.L.P. Chen. Semantic video model for content-based retrieval. In *IEEE International Conferene on Multimedia Computing and Systems*, volume 2, pages 472–478, 1999.

[2] J.Y. Zhou, E.P. Ong, and C.C. Ko. Video object segmentation and tracking for content-based video coding. In *IEEE International Conference on Multimedia and Expo*, volume 3, pages 1555–1558, 2000.

[3] D.A. Adjeroh, M.C. Lee, and I. King. A distance measure for video sequence similarity matching. In *International Workshop on Multi-Media Database Management Systems*, pages 72–79, 1998.

[4] R. Lienhart, W. Effelsberg, and R. Jain. Visualgrep: A systematic method to compare and retrieve video sequences. In *Storage and Retrieval for Image and Video Databases VI, SPIE*, volume 3312, page 271, Jan 1998.

[5] R. Mohan. Video sequence matching. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 6, pages 3697–3700, 1998.

[6] Y. Rui, T.S. Huang, and S. Mehrotra. Constructing table-of-content for videos. *ACM Multimedia Systems Journal, Special Issue Multimedia Systems on Video Libraries*, 7(5):359–368, Sept 1999.

[7] J.S. Boreczky and L.A. Rowe. Comparison of video shot boundary detection techniques. *Journal of Electronic Imaging 5*, pages 122–128, Apr 1996.

[8] P. Browne, A. Smeaton, N. Murphy, S. Marlow, and C. Berrut. Evaluating and combining digital video shot boundary detection algorithms. In *Irish Machine Vision and ImageProcessing Conference (IMVIP 2000), Belfast, Northern Ireland*, Aug 2000.

[9] M.J. Swain and D.H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[10] M.K. Shan and S.Y. Lee. Content-based video retrieval based on similarity of frame sequence. In *International Workshop on Multi-Media Database Management Systems*, pages 90–97, 1998.