A Graph-based Approach for Modeling and Indexing Video Data

Jeongkyu Lee
Department of Computer Science and Engineering
University of Bridgeport
Bridgeport, CT 06604 U. S. A.
E-mail: jelee@bridgeport.edu

Abstract

In this work, we propose new graph-based data model and indexing to organize and manage video data. To consider spatial and temporal characteristics of video, we introduce a new graph-based data model called Spatio-Temporal Region Graph (STRG). Unlike existing graph-based data structures which provide only spatial features, the proposed STRG further provides temporal features, which represent temporal relationships among spatial objects. The STRG is decomposed into its subgraphs object graphs (OGs) and background graphs (BGs). In addition, a new distance measure, called Extended Graph Edit Distance ($EGED$), is introduced in metric space for matching and indexing. Based on clustering and $EGED$, we propose a new indexing method STRG-Index, which is faster and more accurate. We compare the STRG-Index with the M-tree, which is a popular tree-based indexing method for multimedia data. The STRG-Index outperforms the M-tree in terms of cost and speed.

1. Introduction

With the recent advances in electronic imaging, video devices, computing power, and network technologies, the use of multimedia data in many applications has increased significantly. Some examples of these applications are distance learning, digital libraries, video surveillance systems, and medical videos. As a consequence, there are increasing demands on modeling, indexing and retrieving these data. Video is a medium of communication that delivers more information per second than any other elements of multimedia. However, the complexities of video data and their sheer volume as well as the limitation of current video processing techniques, have restricted progress on video data modeling, indexing and retrieval. To address the challenging problems, we propose a graph-based approach for modeling and indexing video data.

Graph is a powerful tool for pattern representation and classification in various fields [13, 2], such as image processing, video analysis, and biomedical applications. The primary advantage of graph-based representation is that it can represent patterns and relationships among data easily. To take this advantage into video analysis, several studies have proposed the graph-based techniques [14, 18, 5, 9]. In Region Adjacency Graph (RAG) [14, 18], segmented regions and spatial relationships among them are expressed as nodes and edges, respectively. However, RAG cannot represent the temporal characteristic of video which is its representative feature. Also, various graph matching algorithms such as bipartite matching [5] and error-correcting matching [9] have been used in video data. However, the existing graph matching algorithms still require high computational cost, and suffer from low accuracy since they consider only the spatial feature to match video data.

In order to address the aforementioned problems, we first propose a new graph-based video data structure, called *Spatio-Temporal Region Graph* (STRG), which represents spatio-temporal features, and the relationships among the video objects [11, 10]. *Region Adjacency Graph* (RAG) [18, 14] is generated from each frame, and STRG is constructed by connecting RAGs. The STRG is segmented into a number of pieces corresponding to shots for efficiency. Then, each segmented STRG is decomposed into its subgraphs, called *Object Graph* (OG) and *Background Graph* (BG) in which redundant BGs are eliminated to reduce index size and search time. The proposed indexing starts with clustering OGs using Expectation Maximization (EM) algorithm [8] for more accurate indexing. To cluster them, we need a distance measure between two OGs. For the distance measure, we propose *Extended Graph Edit Distance* ($EGED$) because the existing measures are not very suitable for OGs. The $EGED$ is defined in non-metric space for clustering OGs, and it is extended to metric space to compute the key values for indexing. Based on the clusters of OGs and the $EGED$, we propose a new indexing structure *STRG-Index* which provides efficient retrieval.

The contributions of the proposed work are as follows:

- We propose a new video data structure, STRG based on graph representation. It can represent not only spatial features of video objects, but also temporal relationships among them.
- We propose a new distance function, $EGED$ which is defined in both non-metric and metric spaces. Non-metric $EGED$ is used for matching video objects, while metric $EGED$ is used for indexing STRG. It provides more accurate distance measure.
- We propose a new indexing structure, STRG-Index which provides fast and accurate indexing since it uses tree structure and data clustering.

The remainder of this paper is organized as follows. In Section 2, we explain how to construct an STRG and how to decompose STRG into OGs. In Section 3, we introduce the $EGED$ for graph matching, and a model-based clustering algorithm to group similar object graphs. We propose STRG-Index for video data in Section 4. The performance study is reported in Section 5. Finally, Section 6 presents some concluding remarks.

2. STRG Construction

In this section, we describe *Spatio-Temporal Region Graph* and *Object Graph* for video objects.

2.1. Spatio-Temporal Region Graph

For a given video, each frame is segmented into a number of regions using a region segmentation technique. Then, Region Adjacency Graph (RAG) is obtained by converting each region into node, and spatial relationships among regions into edges [14], which is defined as follows:

Definition 1 *Given the n^{th} frame f_n in a video, a Region Adjacency Graph of f_n, $Gr(f_n)$, is a four-tuple $Gr(f_n) = \{V, E_S, \nu, \xi\}$, where*

- *V is a finite set of nodes for the segmented regions in f_n,*
- *$E_S \subseteq V \times V$ is a finite set of spatial edges between adjacent nodes in f_n,*
- *$\nu : V \rightarrow A_V$ is a set of functions generating node attributes, and*
- *$\xi : E_S \rightarrow A_{E_S}$ is a set of functions generating spatial edge attributes.*

The node attributes (A_V) represent size (i.e., number of pixels), dominant color and location of corresponding region, the spatial edge attributes (A_{E_S}) represent the relationships between two adjacent nodes such as spatial distance and orientation. RAG is good for representing spatial relationships among nodes indicating the segmented regions. However, it cannot represent temporal characteristics of video. We propose a new graph-based data structure for video, *Spatio-Temporal Region Graph* (STRG) which is temporally connected RAGs [11]. The STRG can handle both temporal and spatial characteristics of video, and defined as follows:

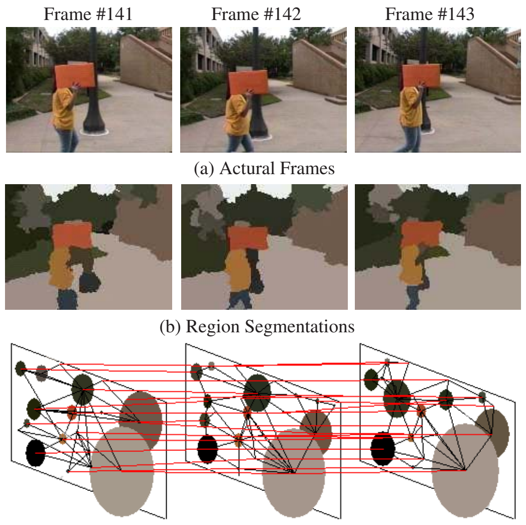
Definition 2 *Given a video segment S, a Spatio-Temporal Region Graph, $Gst(S)$, is a six-tuple $Gst(S) = \{V, E_S, E_T, \nu, \xi, \tau\}$, where*

- *V is a finite set of nodes for segmented regions from S,*
- *$E_S \subseteq V \times V$ is a finite set of spatial edges between adjacent nodes in S,*
- *$E_T \subseteq V \times V$ is a finite set of temporal edges between temporally consecutive nodes in S,*
- *$\nu : V \rightarrow A_V$ is a set of functions generating node attributes,*
- *$\xi : E_S \rightarrow A_{E_S}$ is a set of functions generating spatial edge attributes, and*
- *$\tau : E_T \rightarrow A_{E_T}$ is a set of functions generating temporal edge attributes.*

In STRG, the temporal edge attributes (A_{E_T}) represent the relationships between corresponding nodes in two consecutive frames such as velocity and moving direction. Figure 1 (a) and (b) are actual frames in a sample video and their region segmentation results, respectively. Figure 1(c) shows a part of STRG for frames #141 $-$ #143 constructed by adding temporal edges which are horizontal lines between the frames.

An STRG is an extension of RAGs by adding temporal edges (E_T) to them. E_T represents temporal relationships between corresponding nodes in two consecutive RAGs. The main procedure of building STRG is therefore, how to construct E_T, which is similar to the problem of objects tracking in a video sequence. To find the corresponding nodes in two consecutive RAGs, we use a graph isomorphism and maximal common subgraph [2]. These algorithms are conceptually simple, but have a high computational complexity. To address this, we decompose a RAG into its neighborhood graphs ($G_N(v)$) which are subgraphs of RAG as follows:

Definition 3 *$G_N(v)$ is the neighborhood graph of a given node v in a RAG, if for any nodes $u \in G_N(v)$, u is the adjacent node of v, and has one edge such that $e_S = (v, u)$.*



(a) Actural Frames

(b) Region Segmentations

(c) STRG

Figure 1. Example of a part of STRG

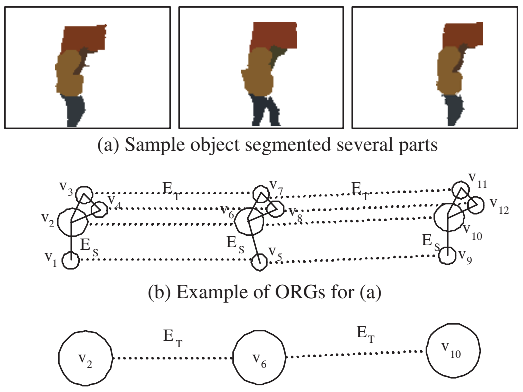
Let $\mathbb{G}_N^m$ and $\mathbb{G}_N^{m+1}$ be sets of the neighborhood graphs in m^{th} and $(m+1)^{th}$ frames respectively. For each node v in m^{th} frame, the goal is to find the corresponding target node v' in $(m+1)^{th}$ frame. To decide these corresponding nodes, we use the neighborhood graphs in Definition 3. For each neighborhood graph $G_N(v)$ in $\mathbb{G}_N^m$, the goal is converted to finding the corresponding target graph $G_N(v')$ in $\mathbb{G}_N^{m+1}$, which is an isomorphic or the most similar graph to $G_N(v)$. First, we find the neighborhood graph in $\mathbb{G}_N^{m+1}$, which is isomorphic to $G_N(v)$. Second, if we cannot find any isomorphic graph in $\mathbb{G}_N^{m+1}$, we find the most similar neighborhood graph to $G_N(v)$ using a similarity measure, $SG(G_N(v), G_N(v'))$, which is defined as follows:

$$SG(G_N(v), G_N(v')) = \frac{|G_C|}{min(|G_N(v)|, |G_N(v')|)} \quad (1)$$

where $|G|$ denotes the number of nodes of G, and G_C is the maximal common subgraph of $G_N(v)$ and $G_N(v')$. G_C can be computed based on maximal clique detection [12]. For $G_N(v) \in \mathbb{G}_N^m$, $G_N(v')$ is the corresponding neighborhood graph in $\mathbb{G}_N^{m+1}$, whose SG with $G_N(v)$ is the largest among neighborhood graphs in $\mathbb{G}_N^{m+1}$, and greater than a certain threshold value. In this way, we find all pairs of corresponding neighborhood graphs (eventually corresponding nodes) from $\mathbb{G}_N^m$ to $\mathbb{G}_N^{m+1}$.

2.2. Object Graph

An STRG is first decomposed into *Object Region Graphs* (ORGs) to model moving objects. We consider a temporal subgraph that can be defined as a set of sequential nodes connected to each other by a set of temporal edges (E_T). An ORG is a special case of temporal subgraph of STRG when the spatial edge set E_S is empty. However, due to the limitations of region segmentation techniques, different color regions belonging to a single object cannot be detected as a single region. For instance, a body of person may consist of several regions such as head, upper body and lower body. Figure 2 (a) shows an object that is segmented into four regions over three frames. Since there are four regions in each frame, we build four ORGs, i.e. (v_1, v_5, v_9), (v_2, v_6, v_{10}), (v_3, v_7, v_{11}), and (v_4, v_8, v_{12}) like Figure 2 (b). Since they belong to a single object, it is better to merge those ORGs into one.



(a) Sample object segmented several parts

(b) Example of ORGs for (a)

(c) Merged OG

Figure 2. The example of OG merging

For convenience, we refer to the merged ORGs as *Object Graph* (OG). In order to merge two ORGs which belong to a single object, we consider the attributes (i.e. velocity and moving direction) of temporal edge (E_T). If two ORGs have same moving direction and same velocity, these can be merged into one. In Figure 2 (c), four ORGs are merged into a single OG, i.e. (v_2, v_6, v_{10}). After OGs are extracted, the remainders of STRG represent background information of a video. We call this graph as a *Background Graph* (BG) that will be used for indexing in Section 4.

3. STRG Clustering

We cluster similar OGs into a group, in which we need to match two OGs. For this graph matching, we introduce a new distance measure, called *Extended Graph Edit Distance* ($EGED$) [10], which can handle temporal characteristics of OGs, then present a clustering algorithm using *Expectation Maximization* (EM).

3.1. Extended Graph Edit Distance

The purpose of the edit distance for graphs is to compute the minimum cost of graph edit operations such as adding, deleting, and changing nodes, to transform one graph to the other. Since the main operations to edit graphs deal with nodes and their attributes rather than edges, we consider only the nodes and their attributes. Let $OG^s_m = \{v^s_1, \ldots, v^s_m, \nu^s\}$ and $OG^t_n = \{v^t_1, \ldots, v^t_n, \nu^t\}$ be s^{th} and t^{th} OGs with m and n number of nodes, respectively.

Definition 4 *The Extended Graph Edit Distance (EGED) between two object graphs OG^s_m and OG^t_n is defined as:*

$$EGED(OG^s_m, OG^t_n) =$$

$$\begin{cases} \sum_{i=1}^{m} |v^s_i - g_i| & \textit{if } n = 1, \\ \sum_{i=1}^{n} |v^t_i - g_i| & \textit{if } m = 1, \\ min[EGED(OG^s_{m-1}, OG^t_{n-1}) + dist(v^s_m, v^t_n), & \\ \quad EGED(OG^s_{m-1}, OG^t_n) + dist(v^s_m, gap), & \\ \quad EGED(OG^s_m, OG^t_{n-1}) + dist(gap, v^t_n)] & \\ & \textit{otherwise.} \end{cases}$$

where gap is an added, deleted or changed node, and g_i is a gap for i^{th} node. And,

$$dist(v^s_i, v^t_j) = \begin{cases} |v^s_i - v^t_j| & \textit{if } v^s_i, v^t_j \textit{ are not a gap} \\ |v^s_i - g_j| & \textit{if } v^t_j \textit{ is a gap} \\ |v^t_j - g_i| & \textit{if } v^s_i \textit{ is a gap.} \end{cases}$$

For better readability, let v indicate a value $\nu(v)$ of node attribute. $dist$ is the cost function for editing nodes. Depending on how to select a gap (g_i), various distance functions are possible. For example, when $g_i = v_{i-1}$, the cost function is the same as one in DTW, which does not consider local time shifting. In our case, $g_i = \frac{v_{i-1}+v_i}{2}$ is used for $dist$, which can handle local time shifting [6].

However, as long as the cost function $dist$ replicates the previous nodes, $EGED$ is no longer in metric space since $dist$ does not satisfy the triangle inequality. In order to satisfy the triangle inequality, $EGED$ is specialized to be metric distance function (see Theorem 1) by comparing the current value with the fixed constant.

Theorem 1 *If g_i is a fixed constant, then EGED is a metric.*

Theorem 1 can be proved by using a mathematic induction. Due to lack of space, we omit a detailed proof. $EGED_M$ is used to indicate the metric version of $EGED$.

3.2. Clustering with EM + $EGED$

In order to group similar OGs, we employ a model-based EM clustering algorithm [3] with $EGED$. Given M mutually independent sample OGs, $\mathcal{Y} = \{y_1, \ldots, y_M\}$, the d-dimensional Gaussian mixture density with $EGED$ is given by

$$p(Y_j|\Theta) = \sum_{k=1}^{K} \frac{w_k}{2\pi^{1/2}|\sigma_k|} e^{-\frac{1}{2\sigma^2} EGED(Y_j, \mu_k)^2} \tag{2}$$

where K is the number of clusters, and w_k ($\sum_{k=1}^{K} w_k = 1$) is the membership probability of the k^{th} cluster. The log-likelihood ($\mathcal{L}$) of K mixture model is

$$\mathcal{L}(\Theta|Y) = log \prod_{j=1}^{M} p(Y_j|\Theta) = \sum_{j=1}^{M} \log \sum_{k=1}^{K} w_k p_k(Y_j|\theta_k) \tag{3}$$

where each θ_k is the set of parameters of the k^{th} cluster. In order to find appropriate clusters, we estimate the optimal values of the parameters (θ_k) and the weights (w_k) in Equation (3) using EM, which is a common procedure used to find the Maximum Likelihood Estimates (MLE) of the parameters iteratively. The EM produces the MLE of the unknown parameters using two alternating steps:
E-step: It evaluates the posterior probability of y_j, belonging to each cluster k. Let h_{jk} be the probability of the j^{th} OG for a cluster k, then it can be defined as follows:

$$h_{jk} = P(k|Y_j, \theta_k) = \frac{w_k}{p_k(Y_j|\theta_k)}$$

M-step: It computes the new parameter value that maximizes the probability using h_{jk} in E-step as follows:

$$w_k = \frac{1}{M}\sum_{j=1}^{M} h_{jk}, \quad \mu_k = \frac{\sum_{j=1}^{M} h_{jk} Y_j}{\sum_{j=1}^{M} h_{jk}}$$

$$\sigma_k = \frac{\sum_{j=1}^{M} h_{jk} EGED(Y_j, \mu_k)^2}{\sum_{j=1}^{M} h_{jk}}$$

The iteration of E and M steps is stopped when w_k is converged for all k. After the maximum likelihood model parameters ($\hat{\Theta}$) in Equation (3) are decided, each OG is assigned to a cluster.

4. STRG Indexing

In this section, we propose a graph-based video indexing method, called *Spatio-Temporal Region Graph Index* (STRG-Index), which uses the $EGED_M$ as a distance measure in metric space, and clustered OGs. We illustrate the structure and construction of STRG-Index, and discuss its search algorithm.

4.1. STRG-Index Tree Structure

To build an index for video data, we adapt the procedure of tree construction proposed in M-tree [7] since it has a minimum number of distance computations and a good I/O performance. In M-tree, a number of representative data items are selected for efficient indexing. There are several ways to select them such as Sampling or Random selection. In the STRG-Index, we employ the clustering results to determine the representative data items. The STRG-Index tree

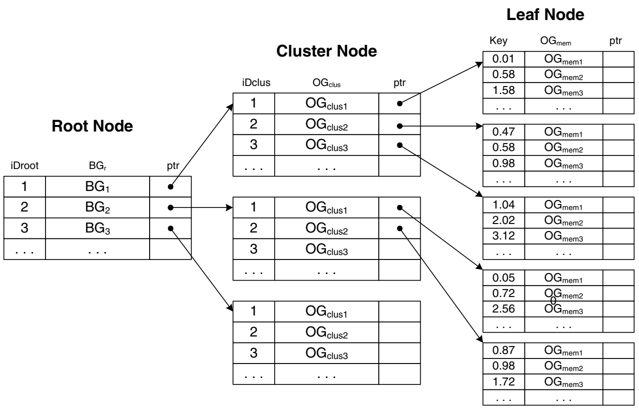


Figure 3. Example of STRG-Index tree structure

structure consists of three levels of nodes; shot node, cluster node, and object node as seen in Figure 3.

The top-level has the shot node which contains the information of each shot in a video. Each record in the shot node represents a segmented shot whose frames share a background. The record has a shot identifier ($ShotID$), a key RAG (Gr_{key}), an actual BG (BG_r), and an associated pointer (ptr) which references the top of corresponding cluster node. The following figure shows an example of a record in the shot node.

ShotID	*Gr_{key}*	*BG_r*	*ptr*
1	RAG_1	BG_1	●→

The mid-level has the cluster nodes which contain the centroid OGs that represent cluster centroids. Each record indicates a representative OG among a group of similar OGs. A record contains its identifier ($ClusID$), a centroid OG (OG_c) of each cluster, and an associated pointer (ptr) which references the top of corresponding object node. The following figure shows an example of a record in a cluster node.

ClusID	*OG_c*	*ptr*
1	OG_{clus1}	●→

The low-level has the object nodes which contain OGs belonging to a same cluster. Each record in the object node represents an object in a video, and has the index key (which is computed by $EGED_M(OG_m, OG_c)$), an actual OG (OG_m), and an associated pointer (ptr) which references the actual video clip in the disk. The following figure shows an example of a record in the object node.

Key ($EGED_M$)	*OG_m*	*ptr*
0.01	OG_{mem1}	●→

4.2. STRG-Index Tree Construction

Based on the STRG decomposition described in Section 2.2, an input video is separated into foreground (OG) and background (BG) as subgraphs of the STRG. The extracted BGs are stored at the root node without any parent. All the OGs sharing one BG are in a same cluster node. This can reduce the size of index significantly. For example, in surveillance videos a camera is stationary so that the background is usually fixed. Therefore, only one record (BG) in the shot node is sufficient to index the background of the entire video.

We synthesize a centroid OG (OG_c) for each cluster which is a representative OG for the cluster. This centroid OG is inserted into an appropriate cluster node as a record. This centroid OG is updated as the member OGs are changed such as inserting, deleting, etc. Also, each record in a cluster node has a pointer to an object node.

The object node has actual OGs in a cluster, which are indexed by $EGED_M$. To decide an indexing value for each OG, we compute $EGED_M$ between the representative OG (OG_c) in the corresponding cluster and the OG (OG_m) to be indexed. Since $EGED_M$ is a metric distance by Theorem 1, the value can be the key of OG to be indexed.

4.3. Search Algorithm

Once the STRG-Index is constructed, we can search and retrieve videos using object-based search. For the object-based search, we extract the background graph BG_q and the object graphs OG_qs from a query video segment ***q*** using the procedure described in Section 2.2. We employ k-Nearest Neighbor (k-NN) search algorithm [17]. At the query time, the BG_q is compared to each record in the shot node to find a matching background and a cluster node that it points to. Then OG_q is compared to the records in the cluster node using $EGED(OG_q, OG_c)$ to find a matching OG_c and an object node that it points to. Finally, the search algorithm travels the object node to find similar OGs by comparing $EGED_M(OG_q, OG_c)$ to the index key values. Algorithm 1 outlines the pseudocode of object-based search algorithm using the k-NN search. In case a query does not consider a background, Step 2 of Algorithm 1 can be skipped, and the search algorithm travels all cluster nodes in STRG-Index to find the similar centroid OGs (OG_c).

Algorithm 1: Object-based k-NN Search Algorithm

```
Input: a query example $S_q$, a STRG-Index, $k$
Output: the $k$ most similar $OGs$

1: extract $OG_q$ and $BG_g$ from $S_q$;
2: find the $BG$ for $BG_q$ in STRG-Index shot node using $TGC$;
3: find the cluster $OG_c$ for $OG_q$ in STRG-Index using $EGED$;
4: $d = EGED_M(OG_q, OG_c)$;
5: find k-nearest neighbor $OGs$ to $OG_q$ using STRG-Index
   Leaf node key and $d$;
6: return $OGs$;
```

5. Experimental Results

To assess the proposed method for clustering OGs, we performed the experiments with the real video streams captured by a video camera. Table 1 shows the description of the video and results of the experiments. The first two videos (Room1 and Room2) were taken from a room in a building, and the other two (Car1 and Car2) from outside, which have some traffic scenes. The third and the fourth columns of Table 1 are the number of actual video objects and the number of correctly detected OGs, respectively. As seen in the fifth column, the accuracy of generating OG reaches up to 94.7% on average.

Table 1. Results of STRG construction and clustering for real video streams

Video	Duration	OG performance			Clustering Error Rate		
		Actual OGs	Found OGs	Accuracy	EM	KM	KHM
Room1	40h 30m	438	411	93.8%	16.8%	33.6%	29.1%
Room2	4h 12m	159	147	92.5%	14.4%	28.9%	22.7%
Car1	15m	202	195	96.5%	8.8%	17.6%	13.0%
Car2	12m	210	203	96.7%	9.5%	17.7%	13.3%
Total	45h 7m	1009	956	94.7%	12.4%	24.5%	19.5%

We compare the performance of EM clustering algorithm with K-means (KM) and K-harmonic means (KHM). To be fair, all of the clustering algorithms use $EGED$. To evaluate the clustering algorithm, we use the clustering error rate defined as:

$$Clustering\ Error\ Rate\ (\%)\ = (1 - \frac{Number\ of\ Correctly\ Clustered\ OGs}{Number\ Of\ Total\ OGs}) \times 100$$

Table 1 also shows that EM is around two times better than KM and KHM in terms of the clustering error rate.

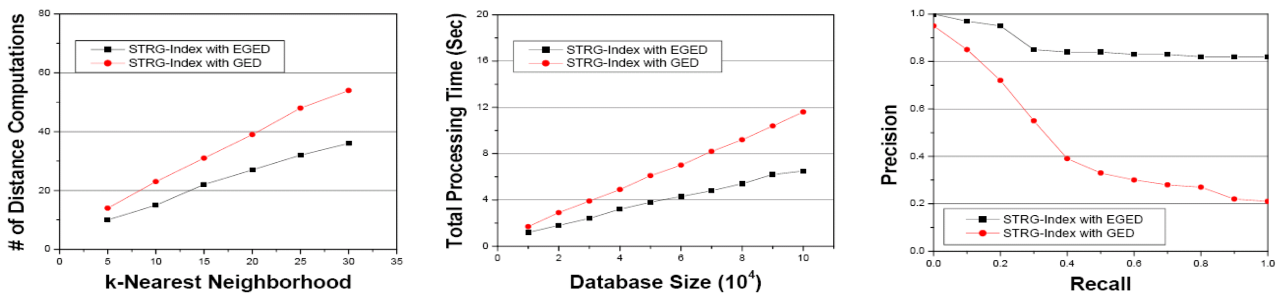
To demonstrate the performance of the proposed STRG-Index, synthetic data is generated and used for the experiments. Since an OG is a type of time-series data, we generate new data by combining the Pelleg data set [15] which is widely used to test clustering algorithms, with the Vlachos data set [16] which is 2-D time-series data with noises.

$EGED$ vs. Graph Edit Distance

In order to evaluate the effectiveness of STRG-Index and $EGED_M$, we first compare the performances of two versions of STRG-Index, which are using $EGED_M$ and the classical edit distance. For the classical edit distance, we use Bunke's Graph Edit Distance (GED) with the simple cost function used in [1], where all the costs of editing nodes are set to one. Since the GED still obeys the triangular inequality [6], it is in a metric space. Therefore, we cluster OGs, and build an STRG-Index by replacing $EGED_M$ with GED. We compare the k-NN query performance of STRG-Index using $EGED_M$ with GED by considering the number of distance computations and the total processing time. Since the number of distance computations performed during a query processing is the dominant component [4], we consider it for evaluating the performance of k-NN query. k neighbors range from 5 to 30 on the synthesized data set which contains 5×10^4 objects in the 480 clusters. Figure 4 (a) shows that the number of distance computations for $EGED_M$ is much smaller (average 30%) than that for GED. However, when the database size increases, the entire STRG-Index may not be fit to the memory. Therefore, we perform 10-NN queries using the STRG-Index on the data sets with various sizes ranging from 1×10^4 to 10×10^4 objects. Figure 4 (b) shows the total processing time which includes the distance computations and the disk I/Os for 10-NN queries. It shows that the total processing time for $EGED_M$ is less than that for GED. Figure 4 (c) shows the accuracy of each indexing for the 10-NN query on a data set with 5×10^4 objects. In order to measure the accuracy, the precision and the recall of query results are computed and plotted. The query data is composed of OGs that are not in the data sets, and the query results are evaluated by the cluster memberships. From Figure 4 (c), it is obvious that the STRG-Index using $EGED_M$ outperforms the STRG-Index using GED, since GED cannot handle the time characteristic of OGs appropriately. Overall, the STRG-Index is more effective when it uses $EGED_M$.

STRG-Index vs. M-tree

Next, we compare STRG-Index with M-tree (MT) index based on cost and accuracy. In the MT indexing, there are several possibilities depending on the criteria used to select the representative data items. RANDOM (MT-RA) and SAMPLING (MT-SA) methods are chosen for comparison purpose since MT-RA is the fastest, and MT-SA is the most



(a) Distance computation for k-NN queries (b) Total processing time for 10-NN queries (c) Accuracy by precision and recall

Figure 4. Query performances of STRG-Index with EGED$_M$ vs. STRG-Index with GED

accurate among the methods proposed in [7]. MT-RA selects the reference object(s) randomly. In STRG-Index, we use the EM clustering for selecting the representative nodes (cluster nodes). Actually, we cannot use the $EGED$ for the MT since it needs a metric distance and does not use any explicit clustering. Therefore, we use the $EGED_M$ for the MT construction where RANDOM or SAMPLE is used for selecting representative nodes. In STRG-Index, we use $EGED_M$ for indexing, and EM clustering for selecting the cluster nodes.

In order to validate the quality of the STRG-Index structure, we perform k-NN queries on the same synthesized data set used in the above experiments. As seen in Fig. 5 (a), the number of distance computations to process k-NN queries using the STRG-Index is much smaller (average 22%) than that using either the MT-RA or the MT-SA. Figure 5 (b) shows the total processing time of 10-NN queries on the data sets ranging from 1×10^4 to 10×10^4 objects. The total processing time for 10-NN query using an STRG-Index is similar to that using MT-SA, and much less than that using MT-RA. This means that the performance of k-NN query using the STRG-Index is better than that using the MT index since both STRG-Index and MT use the same distance measure $EGED_M$. Figure 5 (c) shows the accuracy of each indexing structure for the k-NN query. As seen in the figure, the STRG-Index outperforms both MT-RA and MT-SA. These results demonstrate that the STRG-Index outperforms the M-tree index in terms of both cost and accuracy.

Efficiency and Scalability of STRG-Index

Figure 6 (a) shows the average time elapsed in building an index structure for databases of different sizes. From this figure, the time to build a STRG-Index is much less (15% to 50%) than that to build either MT-RA or MT-SA, even though both STRG-Index and MT have a similar tree structure. The complexity of building the STRG-Index is same as that of clustering because the index structure is built during the clustering process. However, the MT uses a split procedure during the index construction, which takes more time.

Concerning the scalability of STRG-Index with respect to the number of clusters, we compare the size of scalable STRG-Index, where actual OGs are removed, with that of the STRG-Index as the number of clusters increases from 48 to 480. Each cluster has 1,000 objects, which means the total number of objects increases from 4.8×10^4 to 4.8×10^5. As seen in Figure 6 (b), the sizes of both scalable STRG-Index and general STRG-Index scale linearly with respect to the number of clusters, but the size of scalable STRG-Index is much smaller than that of general STRG-Index. This addresses the problem when the database size increases arbitrarily large.

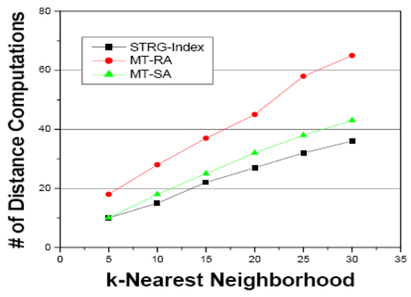
6. Concluding Remarks

In this work we propose a new graph-based data model, spatio-temporal region graph (STRG) representing spatial and temporal relationships among objects in a video. After an STRG is constructed, it is decomposed into object graphs (OGs) and background graph (BG). For unsupervised learning, we cluster similar OGs into a group. In addition, extended graph edit distance ($EGED$) is introduced for graph matching. The $EGED$ is defined on metric spaces and used for indexing key values. Using clustered OGs, we propose a graph-based video indexing method, called STRG-Index. Experimental results on both synthetic data and real video data show the effectiveness and accuracy of the proposed approach.

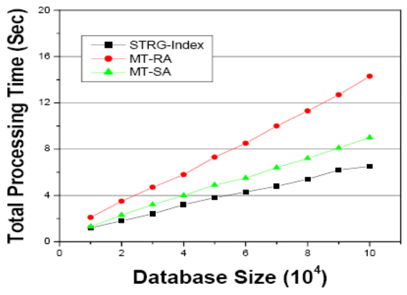
References

[1] H. Bunke. On a relation between graph edit distance and maximum common subgraph. *Pattern Recogn. Lett.*, 18(9):689–694, 1997.

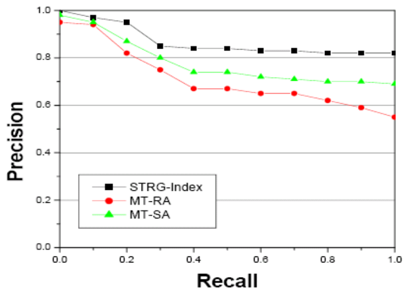
[2] H. Bunke and K. Shearer. A Graph Distance Metric based on the Maximal Common Subgraph. *Pattern Recognition Letters 19*, pages 255–259, 1998.



(a) Distance computation for k-NN queries

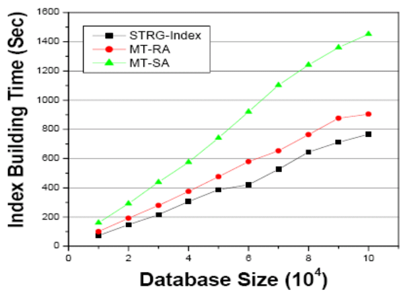


(b) Total processing time for 10-NN queries

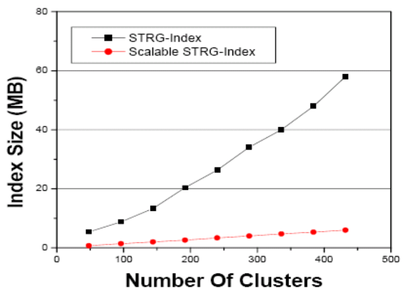


(c) Accuracy by precision and recall

Figure 5. Query performances of STRG-Index comparing with MT-RA and MT-SA



(a) Building index time



(b) Scalability of STRG-Index

Figure 6. Efficiency and Scalability of STRG-Index

[3] R. Chandramouli and V. K. Srikantam. On mixture density and maximum likelihood power estimation via expectation-maximization. In *Proceedings of the 2000 conference on Asia South Pacific design automation*, pages 423–428. ACM Press, 2000.

[4] E. Chavez, G. Navarro, R. Baeza-Yates, and J. L. Marroquin. Searching in metric spaces. *ACM Comput. Surv.*, 33(3):273–321, 2001.

[5] H. T. Chen, H. Lin, and T. L. Liu. Multi-object tracking using dynamical graph matching. *Proc. of the 2001 IEEE Conf. on CVPR*, pages 210–217, 2001.

[6] L. Chen and R. Ng. On the marriage of lp-norms and edit distance. In *The 30th VLDB Conference*, pages 1040–1049, Toronto, Canada, 2004.

[7] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *the VLDB Journal*, pages 426–435, 1997.

[8] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood estimation from incomplete data via the em algorithm (with discussion). *J.R. Statist. Soc. B*, 39:1–38, 1977.

[9] C. Gomila and F. Meyer. Tracking Objects by Graph Matching of Image Partition Sequences. *Proc. of 3rd IAPR-TC15 Workshop on GRPR*, pages 1–11, 2001.

[10] J. Lee, J. Oh, and S. Hwang. Clustering of Video Objects by Graph Matching. In *Proceeding of the 2005 IEEE ICME*, pages 394–397, Amsterdam,Netherlands, July 2005.

[11] J. Lee, J. Oh, and S. Hwang. STRG-Index: Spatio-Temporal Region Graph Indexing for Large Video Databases. In *Proceedings of the 2005 ACM SIGMOD*, pages 718–729, Baltimore, MD, June 2005.

[12] G. Levi. A Note on the Derivation of Maximal Common Subgraphs of Two Directed or Undirected Graphs. *Calcols 9*, pages 341–354, 1972.

[13] S. Lu, M. Lyu, and I. King. Video Summarization by Spatial-Temporal Graph Optimization. In *Proceedings of the 2004 International Symposium on Circuits and Systems*, volume 2, pages 197–200, Vancouver, Canada, May 2004.

[14] O. Miller, E. Navon, and A. Averbuch. Tracking of moving objects based on graph edges similarity. *Proceedings of the ICME '03*, pages 73–76, 2003.

[15] D. Pelleg and A. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 727–734, San Francisco, 2000.

[16] M. Vlachos, D. Gunopulos, and G. Kollios. Robust similarity measures for mobile object trajectories. In *Proc. of 5th International Workshop Mobility In Databases and Distributed Systems*, pages 721–728, Aix-en-Provence, France, September 2002.

[17] C. Yu, B. C. Ooi, K.-L. Tan, and H. V. Jagadish. Indexing the Distance: An Efficient Method to KNN Processing. In *VLDB*, pages 421–430, 2001.

[18] C. Yuan, Y.-F. Ma, and H.-J. Zhang. A Graph-Theoretic Approach to Video Object Segmentation in 2D+t Space. Technical report, MSR, March 2003.