Journal of Visual Languages and Computing (2001) **12,** 627–656
doi:10.1006/jvlc.2000.0222, available online at http://www.idealibrary.com on IDEAL®

Composition and Retrieval of Visual Information for Video Databases

PU-JIEN CHENG AND WEI-PANG YANG*

Institute of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C.
{pjcheng, wpyang}@cis.nctu.edu.tw

Received 9 September 1999; revised 1 June 2000; accepted 22 November 2000

This paper presents a new visual aggregation model for representing visual information about moving objects in video data. Based on available automatic scene segmentation and object tracking algorithms, the proposed model provides eight operations to calculate object motions at various levels of semantic granularity. It represents trajectory, color and dimensions of a single moving object and the directional and topological relations among multiple objects over a time interval. Each representation of a motion can be normalized to improve computational cost and storage utilization. To facilitate query processing, there are two optimal approximate matching algorithms designed to match time-series visual features of moving objects. Experimental results indicate that the proposed algorithms outperform the conventional subsequence matching methods substantially in the similarity between the two trajectories. Finally, the visual aggregation model is integrated into a relational database system and a prototype content-based video retrieval system has been implemented as well.
© 2001 Academic Press

Keywords: content-based retrieval, video data modeling, spatio-temporal composition, query by example and trajectory matching.

1. Introduction

WITH THE PROGRESS of high-capacity storage, high-performance compression and telecommunication technologies, large-scale video archives have been extensively applied. Generally, the content of video data can be classified as *semantic content*, *visual content* and *audio content* [1]. Regarding visual content, video data simultaneously have spatial and temporal characteristics. For temporal features, a visual query may contain either a single image (key-frame) or a continuous sequence of frames. For spatial features in a frame, a visual query may cover either a whole image (global feature) or local sub-images (local feature). Hence, video retrieval systems fall into four categories based on the criterion of spatio-temporal processing granularity: *Single Image Global Feature* (*SIGF*), *Single Image Local Feature* (*SILF*), *Multiple Images Global Feature* (*MIGF*), and *Multiple Images Local Feature* (*MILF*).

The SIGF approach serves to index the whole key-frames extracted from video data with key-frame selection algorithms [2]. Meanwhile, the SILF approach partitions

*Corresponding author.

a key-frame into a set of small regions/objects [3] and analyzes spatial relationships between objects. Both of these approaches are primarily derived from conventional image database technologies, which emphasize visual features of an object or image such as color, texture and shape. The MIGF approach is different; instead of analyzing still images it takes into account a continuous frame sequence within a story unit (scene or shot) such as [4]. The MILF approach goes even further, focusing on the variance in the visual features of local objects over a period of time such as [5–16]. The MILF is the most complicated mechanism and can represent the richness of video content. Its applications include sports event analysis [6], traffic control and surveillance [9, 14].

A complete video data model can be viewed as a layered structure consisting of a low-level physical/audio-visual model and a high-level logical/semantic model. In [1], we have proposed a semantic inference model and annotation language based on the properties of the individual semantic video-object. This investigation adopts the MILF approach to propose an innovative visual aggregation model for expressing spatio-temporal and visual information about moving objects in a video-frame sequence. The proposed model supports numerous composition operations to assemble moving objects at various levels of semantic granularity. The underlying premise of the model is a so-called *motion composition*, a term with a broad range of meanings. For example, a motion composition may describe the physical features of a car crashing into a train or a set of red pixels hopping onto the upper part of a frame. The visual features of moving objects considered herein include trajectory, color and size of a single moving object and directional and topological relations among multiple ones. Size is used to capture the movement of an object along the direction of depth. A user can specify the change of one or more visual features by drawing one or more example motion.

Based on the conventional scene segmentation and object tracking algorithms, a hierarchical structure of motion compositions is constructed semi-automatically and a normalization mechanism is defined to compact each motion composition, thus reducing coding redundancy. To measure the similarity between an example motion and a database motion, this study develops new motion-matching strategies in the form of triangulate string matching algorithms with/without merging. A series of simulation experiments verifies the effectiveness of the proposed algorithms. The final stage is to integrate the proposed conceptual model into a relational database system and implement a prototype content-based video retrieval system.

The rest of this paper is organized as follows. Section 2 describes related work in content-based video retrieval by motion example. Section 3 introduces a new visual aggregation model and corresponding composition operations. Section 4 designs two matching algorithms for measuring the distance between two object motions. Section 5 presents experiments to evaluate the performance of the proposed matching algorithms. Section 6 describes the prototype system's storage structures, and Section 7 contains some concluding remarks.

2. Related Work

Spatio-temporal information (Multiple Images Local Feature, MILF) analysis has been increasingly applied to video data in recent years [5–18]. In contrast to semantic content analysis, MILF allows automatic video content processing and the accessing of video data by specifying an example query. The conventional approach to MILF is to first partition raw video into basic units (i.e. scenes/shots), recognize and track video objects in a scene, extract visual features from any moving objects and finally model and index these time-series visual features of moving objects. To compare two video sequences in terms of the difference in object motion, a subsequence matching algorithm is required to estimate their degree of similarity.

So far, numerous scene detection techniques [19] have been successfully developed. MPEG-4 has standardized object-based video representation and compression, making object recognition and tracking more efficient [20]. However, finding a general solution to automatically extract and group objects of interest is very difficult [7, 9] since the mapping between high- and low-level semantic features is complicated. In addition to the automatic segmentation of specific objects under some constraints, certain operations are required to group these fragmented objects into a meaningful one. These operations not only allow multiple objects to be assembled into higher semantic abstraction recursively but also provide the multiplicity of views for spatio-temporal layout of the moving objects in a scene [21]. To the best of our knowledge, no previous research has attempted to develop an efficient mechanism to tackle this problem.

Several annotation structures have been developed to represent spatio-temporal features (e.g. trajectory) corresponding with a single object in the form of chain codes [12, 16] or vertex sequences [7, 17]. These methods fail to describe the spatial relations among multiple moving objects. The relevant aspects of a moving object include trajectory (direction and displacement), color, shape, size, velocity, acceleration, etc. [12, 16] merely consider the direction of a trajectory and omit the displacement. Although [13] incorporates the displacement into its representation, it lacks matching algorithms to evaluate this displacement. VideoQ [7] supports a rich set of visual features but most of them remain consistent during an object's movement. On the other hand, some GIS and image database technologies have been applied to represent directional and topological relations [22], 2D [5], 2D-B [15] and 2D-C string [11], which are limited in terms of satisfactorily representing a single object's time-series visual features. [13] is the first to concurrently represent the trajectory of a moving object and the relative spatio-temporal relations between moving objects. Unfortunately, its model does not describe other visual features such as color and size.

In addition to the above limitations, when they are given an example of a moving object, the conventional systems [7, 14, 16] produce a candidate list of matched objects for each visual feature. All the candidate lists are then merged to get the final ordering of candidate objects. The conventional systems thus neglect the relative ordering of different feature sequences.

3. Object Motion Analysis

A *video stream v* is an ordered set of frames $\{I_1, I_2, I_3, \ldots, I_N\}$, where I_k is the kth frame in the sequence. For simplicity, we assume each frame is an $R_X \times R_Y$ image. A *scene s* is defined as a video subsequence $\{I_{s.\alpha}, I_{s.\alpha+1}, \ldots, I_{s.\beta}\}$, where $1 \leq s.\alpha \leq s.\beta \leq N$. A (*visual*) *object* either meaningful or meaningless to users is a physical region in a frame such as a face, car and red circle. τ denotes the set of objects that appear in v. An object comprises a set of pixels and several objects can constitute a composite one by means of set operations. Each object contains two types of visual features: *mean features* and *derived features*. Mean features can be obtained by averaging the time-series visual features of the object such as direction, color and sizes while accumulating time-series features can get derived features such as displacement or length. The functions used herein are defined as follows.

λ is a function mapping from $\nu \times \tau$ to £[, where £[is the set of all possible $R_X \times R_Y$ binary images. For each $\lambda(I_i, o) = b_i^o$ $(o \in \tau)$, $b_i^o(x,y)$ indicates the value of the pixel at coordinate (x,y) of the bitmap b_i^o. $b_i^o(x,y)$ is set to be 1 if the object o covers the location (x,y) on I_i; otherwise, $b_i^o(x,y)$ is set to be 0. An *interval* $[x,y]$ represents a continuous sequence (ordered set) of numbers from x to y $(x \leq y)$. The size of the interval $[x,y]$ is $y - x + 1$, denoted by $\|[x,y]\|$. $[x_1,y_1] < [x_2,y_2]$ if and only if $y_1 < x_2$. Given an interval set $T = \{[x_i,y_i] \mid i = 1, .., n\}$, the disjoint composition of T, denoted by *disjoin*(T), is a set of (disjoint) intervals. For each $[x',y'] \in$ *disjoin*(T), the following hold:

1. $\exists i([x',y'] \subseteq [x_i,y_i])$,
2. $\forall i([x',y'] \subseteq [x_i,y_i] \vee [x',y'] \cap [x_i,y_i] = \emptyset)$ and
3. $\neg\exists [x'',y''] ([x'',y''] \in disjoin(T) \wedge [x',y'] \subset [x'',y''])$.

$\vee$ and $\wedge$ stand for logical OR and AND, respectively.

For example, *disjoin*({[1,4], [3,7], [9,10]}) = {[1,2], [3,4], [5,7], [9,10]}.

3.1. The Visual Aggregation Model

The model includes three major data structures: feature, motion and relation sequences.

Definition 3.1. A *feature sequence f* is a four-tuple $f = (s, [\alpha, \beta], o, \psi)$ which represents the feature value of the object o in a scene subsequence $\{I_\alpha, I_{\alpha+1}, \ldots, I_\beta\}$, where α and β are frame numbers of the first and last frames of the feature sequence such that $s.\alpha \leq \alpha \leq \beta \leq s.\beta$, $o \in \tau$ and ψ consists of an ordered set of feature clips $\{c_1^f, c_2^f, \ldots, c_{L_f}^f\}$. A *feature clip* c_i^f stands for a sub-interval of $[s.\alpha, s.\beta]$, denoted by $[c_i^f.\alpha, c_i^f.\beta]$, in which the object o has a constant feature in some feature spaces. Notice that $[c_i^f.\alpha, c_i^f.\beta] < [c_{i+1}^f.\alpha, c_{i+1}^f.\beta]$ and $\bigcup_{i=1,\ldots,L_f}[c_i^f.\alpha, c_i^f.\beta] = [s.\alpha, s.\beta]$. Each feature clip c_i^f is represented as $(\delta, \angle)$, where $c_i^f.\delta (= c_i^f.\beta - c_i^f.\alpha + 1)$ indicates the number of frames and $c_i^f.\angle$ expresses o's *mean-feature* values. No two consecutive feature clips have the same values. Thus, $c_i^f.\angle \neq c_{i+1}^f.\angle$ $(i = 1, \ldots, L_f - 1)$.

To simplify the representation, the elements of a feature sequence f defined in Definition 3.1 are denoted by $f.s$, $f.\alpha$, $f.\beta$, $f.o$ and $f.\psi$, respectively. Similarly, the notation $x.y$, which denotes the value of the element/component y of x, is used hereinafter. For example, $q.s$ represents the scene s where the motion sequence q appears, as shown in Definition 3.2.

Definition 3.2. A *motion sequence q* is a four-tuple $q = (s, [\alpha, \beta], o, \ell)$ which specifies the motion of the object o in a scene subsequence $\{I_\alpha, I_{\alpha+1}, \ldots, I_\beta\}$, where α and β are frame numbers of the first and last frames of the motion sequence such that $s.\alpha \leq \alpha \leq \beta \leq s.\beta$, $o \in \tau$ and ℓ denotes an ordered set of motion clips $\{c_1^m, c_2^m, \ldots, c_{L_m}^m\}$. Similar to a feature clip, a *motion clip* c_i^m is represented by $(\delta, \upsilon, \omega)$, where $c_i^m.\delta$ $(= c_i^m.\beta - c_i^m.\alpha + 1)$ indicates the number of frames, υ specifies an array of m pointers each of which refers to the corresponding feature clip in a feature sequence and ω denotes an array of *derived feature* values of the object o. Notice that the interval set of motion clips is computed by the disjoint composition of that of all feature clips for an object, that is,

$$\bigcup_{\forall c_i^m \in \ell} \{[c_i^m.\,\alpha, c_i^m.\,\beta]\} = disjoin\left(\bigcup_{\forall \psi \forall c_j^f \in \psi} \{[c_j^f.\,\alpha, c_j^f.\,\beta]\}\right).$$

Definition 3.3. A *relation sequence* r is a three-tuple $r = (s, (q_x, q_y), \vartheta)$ specifying the spatial relations of the two objects $q_x.\,o$ and $q_y.\,o$ in the sub-interval $[\alpha', \beta']$ of the scene s, where q_x and q_y indicate motion sequences, $q_x.\,s = q_y.\,s = s$, $q_x.\,o \neq q_y.\,o$, $q_x.\,o, q_y.\,o \in \tau$, $[\alpha', \beta'] = [q_x.\,\alpha, q_x.\,\beta] \cap [q_y.\alpha, q_y.\,\beta]$ and ϑ denotes an ordered set of relation clips $\{c_1^r, c_2^r, \ldots, c_{L_r}^r\}$. A *relation clip* c_i^r stands for a sub-interval of $[\alpha', \beta']$, denoted by $[c_i^r.\,\alpha, c_i^r.\,\beta]$, in which the two objects' mean features and spatial relationship are constant. $[c_i^r.\,\alpha, c_i^r.\,\beta] < [c_{i+1}^r.\,\alpha, c_{i+1}^r.\,\beta]$ and $\bigcup_{i=1,\ldots,L_r} [c_i^r.\,\alpha, c_i^r.\,\beta] = [s.\,\alpha, s.\,\beta]$. Each relation clip c_i^r is represented as (δ, μ, ζ), where $c_i^r.\,\delta (= c_i^r.\,\beta - c_i^r.\,\alpha + 1)$ indicates the number of frames, μ specifies an array of two pointers referring to the corresponding motion clips of the motion sequences q_x and q_y, respectively, and ζ defines the directional and topological relation between them. For arbitrary consecutive relation clips c_i^r and c_{i+1}^r, the rule of $c_i^r.\,\zeta \neq c_{i+1}^r.\,\zeta$ is satisfied.

[23] introduces 13 spatial relations (R_1–R_{13}) between two objects (o_1 and o_2) projected into 1-D space. The 13 relations present directional and topological information together. R_1 and R_{13} indicate 'disjoint', R_2 and R_{12} 'meet', R_3 and R_{11} 'overlap', R_4 and R_8 'cover', R_{10} and R_6 'cover_by', R_5 'contain', R_9 'inside', and R_7 'equal' 1. [23] can be farther extended into 13^2 relations in 2-D space. Herein, we use $[R_i, R_j]$ to describe the 169 relations of the projections on the x- and y-axis. If $[R_i, R_j]$ becomes $[R_i', R_j']$ after at least k moves/transitions [8], k is the distance between them. According to Definition 3.3, a relation clip is a part of a motion clip. Querying arbitrary combinations of the spatial relations and visual features of two objects at the same time becomes possible.

Consequently, a motion (or feature) sequence denotes the motion (or feature) of a visual object. A relation sequence denotes the relations between two objects. A motion (or feature) clip is a segment of a motion (or feature) sequence with different mean features to the preceding and following segments of the sequence. All the frames in a clip share identical mean features. In our system, *mean features* include direction, color and x- and y-axis-projection-interval sizes. *Derived features* include displacement/length.

For example, Figure 1 reveals a scenario of a storm (o_1) approaching, hitting, overlapping and then going away from a land (o_2) in the weather forecast. Let q_1 and q_2 be the motion sequences of the objects o_1 and o_2, respectively. The features of direction (f_1) and size (f_2) are considered herein. o_1's $f_1 = (s_1, [1,510], o_1, \{((330, \text{northwest}), (180, \text{northeast}))\}$. o_1's $f_2 = (s_1, [1,510], o_1, \{(180,220), (330,350)\}$. o_2's $f_1 = (s_1, [1,510], o_2, \{(510,\text{still})\}$. o_2's $f_2 = (s_1, [1,510], o_2, \{(510,380)\}$. $q_1 = (s_1, [1,510], o_1, \{(180,(1,1),(1)), (150, (1,2),(0.4)),(180,(2,2),(0.8))\})$ and $q_2 = (s_1, [1,510], o_2, \{(510, (1,1),(1))\})$. The relation sequence of o_1 and o_2 is $((s_1, (q_1, q_2), \{(150, (1,1), [R_1, R_{11}]), (30,(1,1), [R_2, R_6]), (150,(2,1), [R_3, R_9]), (30,(3,1), [R_3, R_{10}]), (150,(3,1), [R_3, R_3])\})$, as illustrated in Figure 2.

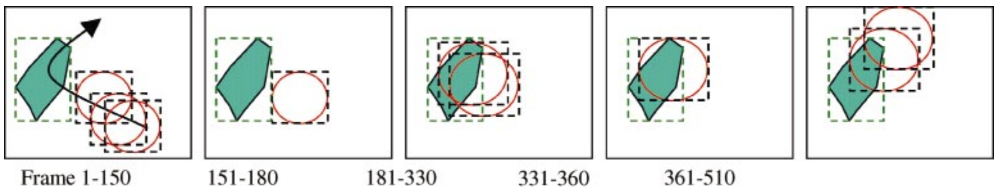


Figure 1. The change of visual information about two objects

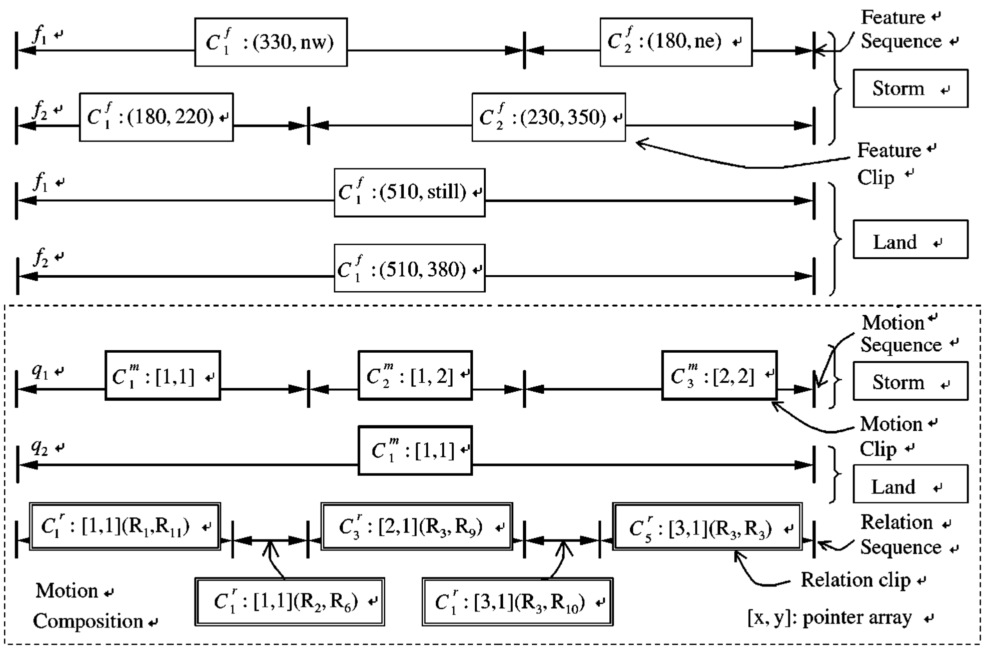


Figure 2. The representation of feature, motion and relation sequences

Definition 3.4. A *motion composition m* is a two-tuple $m = (Q, R)$, where Q and R are, respectively, the sets of *motion sequences* and *relation sequences* satisfying the following rule. For each motion composition, no motion sequences of a single object in the same scene meet or overlap. Thus, if a motion composition contains two motion sequences q_i and q_j such that $q_i.s = q_j.s$ and $q_i.o = q_j.o$, then $[q_i.\alpha, q_i.\beta] \cup [q_j.\alpha, q_j.\beta] \neq [q_i.\alpha, q_j.\beta]$ and $[q_i.\alpha, q_i.\beta] \cap [q_j.\alpha, q_j.\beta] = \emptyset$.

3.2. Feature Extraction

This section discusses how to automatically extract mean and derived feature values from a feature clip c_j^f in a motion sequence q. For a given binary image $b_i^{q.o}$, which describes the location where the object $q.o$ appears on I_i, we assume that Max$X(b_i^{q.o})$, Min$X(b_i^{q.o})$, Max $Y(b_i^{q.})$, and Min $Y(b_i^{q.o})$, respectively, return the object's projections on the x- and y-axis such that its minimum bounding rectangle, $MBR(b_i^{q.o})$, is defined by $((\mathrm{Min}X(b_i^{q.o}), \mathrm{Min}\,Y(b_i^{q.o})), (\mathrm{Max}X(b_i^{q.o}), \mathrm{Max}\,Y(b_i^{q.o}))$. $c_j^f.\alpha$ and $c_j^f.\beta$, frame numbers of the first and last frames of the feature clip c_j^f, are computed as

$$c_j^f.\alpha = q.\alpha + \sum_{t=1}^{j-1} c_t^f.\delta \quad \text{and} \quad c_j^f.\beta = c_j^f.\alpha + c_j^f.\delta - 1. \tag{1}$$

Then, the direction of the feature clip c_j^f in the motion sequence q is computed as

$$Dir(q, c_j^f) = \begin{cases} \dfrac{1}{c_j^f.\delta - 1} \displaystyle\sum_{t=c_j^f.\alpha}^{c_j^f.\beta - 1} (Loc(b_{t+1}^{q.o}) - Loc(b_t^{q.o})) & \text{if } c_j^f.\delta \neq 1 \\ 0 & \text{if } c_j^f.\delta = 1 \end{cases} \tag{2}$$

where

$$Loc(b_t^{q.o}) = ((\mathrm{Max}X(b_t^{q.o}) + \mathrm{Min}X(b_t^{q.o}))/2, (\mathrm{Max}\,Y(b_t^{q.o})) + \mathrm{Min}\,Y(b_t^{q.o}))/2) \tag{3}$$

Since the MBR representation tends to be sensitive to noise, the center of gravity can be computed as the coordinate of an object to reduce the sensitivity. The x-axis-projection-interval size of the feature clip c_j^f in the motion sequence q is evaluated as

$$X - ProjSize(q, c_j^f) = \frac{(1/c_j^f.\delta) \sum_{t=c_j^f.\alpha}^{c_j^f.\beta} (\mathrm{Max}X(b_t^{q.o}) - \mathrm{Min}X(b_t^{q.o}))}{(1/c_1^f.\delta) \sum_{t=c_1^f.\alpha}^{c_1^f.\beta} (\mathrm{Max}X(b_t^{q.o}) - \mathrm{Min}X(b_t^{q.o}))} \tag{4}$$

Similarly, replacing X with Y, respectively, yields the y-axis-projection-interval size.

The color histogram of the feature clip c_j^f in the motion sequence q is formulated as

$$H(q, c_j^f, n) = \frac{1}{c_j^f.\delta} \sum_{t=c_j^f.\alpha}^{c_j^f.\beta} h(I_t, b_t^{q.o}, n), \quad n = 0, \ldots, N_b - 1 \tag{5}$$

where n denotes a bin/color, N_b is the total number of bins, and $h(I_t, b_t^{q.o}, n)$, the ratio of the number of pixels having color n to the number of bits whose values are equal to 1 on $b_t^{q.o}$, is defined as

$$h(I_t, b_t^{q.o}, n) = h'(I_t, b_t^{q.o}, n) \Big/ \sum_{k=0}^{N_b-1} h'(I_t, b_t^{q.o}, k) \tag{6}$$

where

$$h'(I_t, b_t^{q.o}, n) = \sum_{x=0}^{R_N-1} \sum_{y=0}^{R_Y-1} \begin{cases} b_t^{q.o}(x,y) & \text{if } I_t(x,y) = n \\ 0 & \text{otherwise} \end{cases}$$

If the motion clip c_k^m refers to the feature clip c_j^f in the feature space of direction, then the relative length/displacement of the motion clip c_j^m in the motion sequence q is formulated as

$$len^{m}_{c_k^m \to c_j^f}(q, c_k^m) = \frac{c_k^m.\delta}{c_j^f.\delta} \times Len^f(q, c_j^f) \tag{7}$$

where $Len^f(q, c_j^f) = \| Loc(b^{q.o}_{c_j^f.\beta}) - Loc(b^{q.o}_{c_j^f.\alpha}) \| / \| Loc(b^{q.o}_{c_j^f.\beta}) - Loc(b^{q.o}_{c_j^f.\alpha}) \|$ and $\| \; \|$ is L_2 norm.

Equations (4) and (7) derive normal projection-interval size and normal displacement, respectively, to provide a scale for database and query objects. Other features such as velocity and acceleration can be easily integrated into the system.

3.3. Normalization

Since a motion sequence can be broken down into a sequence of interconnected motion clips, the total clip number dominates the required storage space. Compacting/normalizing the motion sequences will significantly reduce computational time and storage space.

A *salient image sequence* is generated according to certain motion features, including the object's color, size and trajectory. Consider a motion sequence q with the duration in terms of frames $\{I_{q.\alpha}, I_{q.\alpha+1}, \dots, I_{q.\beta}\}$. $I_{q.\alpha}$ is treated as the first salient image in default. Let d_i be $Loc(b_i^{q.o})$ as defined in Eq. (3) and $\theta(\vec{x}, \vec{y})$ be the angle between the vector $\vec{x}$ and the vector $\vec{y}$. Figure 3 shows the salient image based on the trajectory, generated as follows:

1. If $min(|\theta(\overrightarrow{d_i d_{i-1}}, \overrightarrow{d_i d_{i+1}})|, 2\pi - |\theta(\overrightarrow{d_i d_{i-1}}, \overrightarrow{d_i d_{i+1}})|) < \sigma_1^{traj}$,then I_i is a salient image.
2. If $min(|\theta(\overrightarrow{d_{i-k} d_{i-k+1}}, \overrightarrow{d_{i-k} d_i})|, 2\pi - |\theta(\overrightarrow{d_{i-k} d_{i-k+1}}, \overrightarrow{d_{i-k} d_i})|) < \sigma_2^{traj}$, where I_{i-k} is a salient image and $I_{i-k+1}, \dots, I_{i-1}$ are not salient images, then I_i is a salient image.

σ_1^{traj} and σ_2^{traj} are two thresholds to detect abrupt and gradual shape changes, respectively.

Similar to trajectory feature, let $e(q, I_i)$ be an evaluation function computing color and projection-interval size of the object $q.o$ in the image I_i as defined in Eqs. (4) and (5), where a motion clip c_j^f contains single frame I_i. $\nabla(e_1, e_2)$ in Figure 4 is a dissimilarity function. The salient image based on the feature of color histogram or projection-interval size is generated as follows:

1. If $|\nabla(e(q, I_{i-1}), e(q, I_i))| > \sigma_1^Z$, then I_i is a salient image.
2. If $|\nabla(e(q, I_{i-k}), e(q, I_i))| > \sigma_2^Z$, where I_{i-k} is a salient image and $I_{i-k+1}, \dots, I_{i-1}$ are not salient images, then I_i is a salient image.

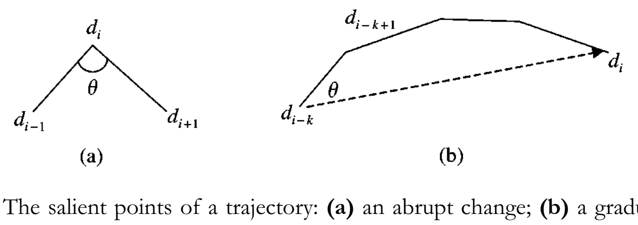


Figure 3. The salient points of a trajectory: **(a)** an abrupt change; **(b)** a gradual change

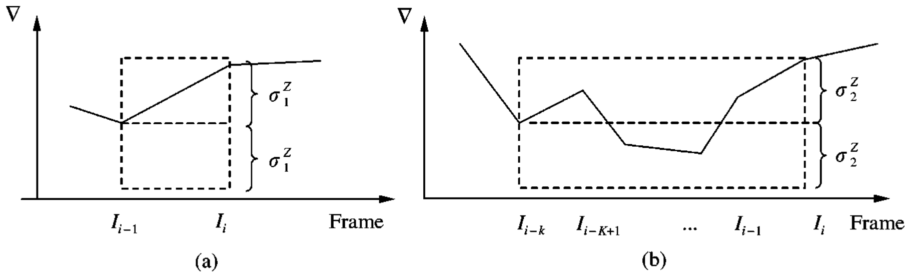


Figure 4. The salient points of the features of color histogram and projection-interval size: **(a)** an abrupt change; **(b)** a gradual change

σ_1^Z and σ_2^Z are two thresholds, where Z stands for either color histogram or projection-interval size. For each visual feature, a local salient image sequence is generated. Eventually, the global salient image sequence of a motion sequence is the union of local salient image sequences based on the features of trajectory, color histogram and projection-interval size. The dissimilarity of color histogram is defined as

$$\nabla^{hist}(e^{hist}(q, I_x), e^{hist}(q, I_y))$$
$$= \sum_{i=0}^{N_b-1} \sum_{j=0}^{N_b-1} a_{ij}(h(I_x, b_x^{q.o}, i) - h(I_y, b_y^{q.o}, i))\,(h(I_x, b_x^{q.o}, j) - h(I_y, b_y^{q.o}, j)) \tag{8}$$

where a_{ij} is the correlation coefficient between color i and color j. The dissimilarity of projection-interval size is computed by

$$\nabla^{size}(e^{size}(q, I_x), e^{\mathrm{size}}(q, I_y)) = \frac{|e^{size}(q, I_y) - e^{size}(q, I_x)|}{\min(e^{size}(q, I_x), e^{\mathrm{size}}(q, I_y))} \times 100\%. \tag{9}$$

Definition 3.5. *Normalization*. For each (mean) feature of a moving object, a feature sequence $f = (s, [\alpha, \beta], o, \{c_1^f, \dots, c_{L_f}^f\})$ and a salient image sequence S are generated. If $I_{C_i^f.\alpha} \in S$ for $i = 1, \dots, L_f$ and $|S| = L_f$, then the feature sequence f is normalized. If all of the feature sequences associated with a motion sequence q are normalized, then the motion sequence q is normalized.

For example, assume that an image sequence demonstrates a person riding a skateboard from left to right, as Figure 5 illustrates. The mean features of trajectory (T), color histogram (C), X-axis-projection-interval size (W) and Y-axis-projection-interval size (H) are extracted for each frame. According to the thresholds of $\sigma_1^{traj} = 135$, $\sigma_2^{traj} = 30$, $\sigma_1^{hist} = \sigma_2^{hist} = 0.45$, $\sigma_1^W = \sigma_2^W = \sigma_1^H = \sigma_2^H = 0.5$, the salient image sequences based on the features of trajectory, color histogram and x- and y-axis-projection-interval size are (I_1, I_4), (I_1, I_8), (I_1, I_2, I_8, I_{10}) and (I_1, I_4, I_8), respectively. The corresponding normalized motion sequence has the interval set of $\{[I_1, I_1], [I_2, I_3], [I_4, I_7], [I_8, I_9], [I_{10}, I_{10}]\}$.

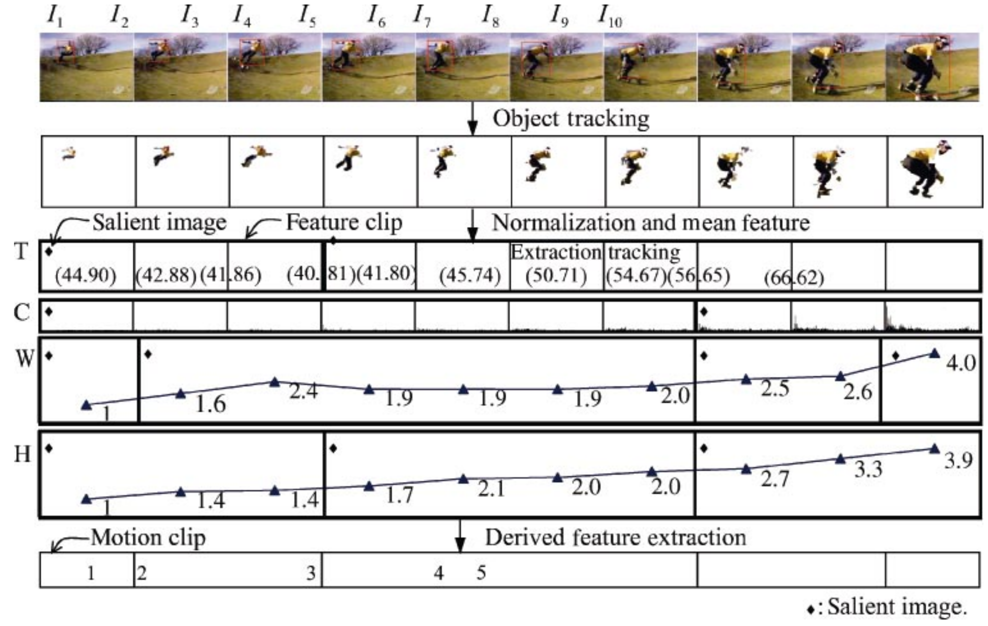


Figure 5. Normalization of a motion sequence

3.4. Composition Operations

This section introduces a series of operations used to manipulate motion compositions. These operations have the property of closure.

The composition operations on the motion compositions are

(1) *Creation and deletion of a motion composition*,
(2) *Selection and projection of a motion composition*, and
(3) *Join and union/intersection/difference of two motion compositions.*

A motion composition includes a set of motion and relation sequences. A motion sequence $q = (s, [\alpha, \beta], o, \ell)$ is mainly determined by $q.s$, $q.\alpha$, $q.\beta$ and $q.o$. $q.\ell$ can be derived from these by applying feature extraction and normalization as outlined in Sections 3.2 and 3.3. Similarly, a relation sequence can be derived from overlapped motion sequences. This section merely emphasizes what $q.s$, $q.\alpha$, $q.\beta$ and $q.o$ are. The following motion database shown in Figure 6 serves as an example. Consider the three motion compositions of $m_1 = \{q_1, q_2, q_3, r_1\}$, $m_2 = \{q_4, q_5\}$, and $m_3 = \{q_6, q_7\}$, where $q_1 = (s_1, [1, 260], o_1, \ell_1)$, $q_2 = (s_1, [100, 450], o_2, \ell_2)$, $q_3 = (s_2, [600, 920], o_1, \ell_3)$, $q_4 = (s_1, [200, 300], o_2, \ell_4)$, $q_5 = (s_2, [800, 1500], o_1, \ell_5)$, $q_6 = (s_1, [500, 630], o_3, \ell_6)$, $q_7 = (s_2, [800, 920], o_4, \ell_7)$ and $r_1 = (s_1, (q_1, q_2), \vartheta_1)$. r_1 describes the spatial relations of o_1 and o_2 in $[100, 260]$.

In Figure 6, a motion composition in our data model can be viewed as a relation in the relational data model. The *selection* operation (α) returns a motion composition consisting of all sequences from a specified motion composition that satisfy a specified condition.

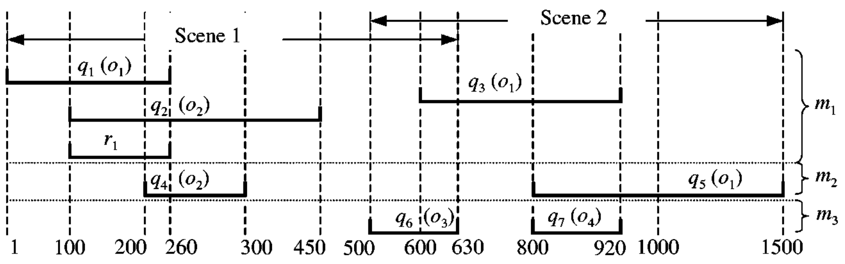


Figure 6. An example motion database with three motion compositions

For example, $\sigma_{\forall q \in m_1.Q(q.s = s_1 \text{ and } a.o = o_1)}(m_1) = \{q_1\}$. The *projection* operation (π) returns a motion composition consisting of all sequences that remain as subsequences in a specified motion composition after specified intervals have been eliminated. For example, $\pi_{[100,200]}(m_1) = \{q_8 = (s_1, [100,200], o_1, \ell_8), q_9 = (s_1, [100,200], o_2, \ell_9), r_2 = (s_1, (q_8, q_9), \vartheta_2)\}$. The *join* operation ($\otimes$) returns a motion composition consisting of all sequences appearing in either or both of two specified motion compositions. Notice that if two returned motion sequences with the same scene and the same object have intersection, they should be merged together. For example, $m_1 \otimes m_2 = \{q_1, q_2, q_{10} = (s_2, [600, 1500], o_1, \ell_{10}), r_1\}$ in which q_4 and q_2 are merged into q_2, and q_3 and q_5 are merged into q_{10}. The *union* ($\cup$), *intersection* ($\cap$), *difference* ($-$) operations are the standard mathematical operations on sets. For example, $m_2 \cup m_3 = \{q_4, q_6, q_{11} = (s_2, [800, 920], o_1 \cup o_4, h_{11}), \quad q_{12} = (s_2,[921, 1500], o_1, \ell_{12})\}$. $m_1 \cap m_2 = \{q_4, q_{13} = (s_2, [800, 920], o_1, \ell_{13})\}$. The intersection operation can be expressed using union and difference as $m_2 \cap m_3 = (m_2 \cup m_3)$-$((m_3 - m_2) \cup (m_2 - m_3))$, where $m_2 \cap m_3 = \emptyset$, $m_2 - m_3 = m_2$ and $m_3 - m_2 = m_3$. The set of composition operations $\{\sigma, \pi, \otimes, \cup, -\}$ is a complete set by which any spatial and temporal combinations as a sequence of operations can be expressed.

Given two motion compositions m_1 and m_2, the complexity of selection/projection of m_1 is $O\left(\sum_{\forall t (t \in m_1.Q \cup m_2.Q \cup m_1.R \cup m_2.R)} t.\delta\right)$, where t is either a motion sequence or a relation sequence and the complexity of join and set operations of m_1 and m_2 is $O\left(|m_1|\log|m_1| + |m_2|\log|m_2| + k \sum_{\forall t (t \in m_1.Q \cup m_2.Q \cup m_1.R \cup m_2.R)} t.\delta\right)$, where $|m_1|$ and $|m_2|$ are, respectively, the size of m_1 and m_2, and k is the mean time of single-image processing.

4. Motion-Matching Algorithm

This section develops new motion-matching algorithms for evaluating the difference between an example motion and a database motion. If each clip is treated as an abstract symbol, the motion-matching problem is equivalent to the approximate string-matching (ASM) problem. Similar to the edit distance between two symbols in string matching, the penalty function to evaluate the dissimilarity between two relation clips c_1^r and c_2^r is defined as

$$p(c_1^r, c_2^r) = \sum_{\forall k (k \in \{traj, color, size\})} \omega_k \cdot \nabla(e_1^k, e_2^k) + \omega_{spatial\_relation} \cdot \nabla(c_1^r.\zeta, c_2^r.\zeta) \geq 0$$

where ω_k is a weighted coefficient of feature k determined by users and ∇ is illustrated in Eqs. (8) and (9). The dissimilarity of trajectory between two motion/relation clips is defined later in Section 5.1. $p(c_1^r, c_2^r)$ is a weighted sum of the dissimilarity between the mean, derived and spatial relation features of c_1^r and c_2^r. The distance between two motion clips can be computed by $p(c_1^r, c_2^r)$ with $\omega_{spatial\_relation} = 0$.

The motion-matching problems can be categorized as full matching and partial matching. Full matching concerns the correspondence of whole sequences while partial matching concerns the match between two subsequences.

4.1. Full Matching

Definition 4.1. *Triangulate association function* (TAF). $TAF_{A \to B}$ (abbreviated as TAF if not confused) is a triangulate association function of sequences $A = a_1a_2a_3, \ldots, a_n$ and $B = b_1b_2b_3, \ldots, b_m$ if and only if there is an onto mapping TAF: $\{1,2,\ldots,n\} \to \{[x,y] \,|\, 1 \leq x,y \leq m\}$ such that $(i<j) \Rightarrow ((TAF(i) = TAF(j) \wedge |TAF(i)| = |TAF(j)| = 1) \vee (TAF(i) < TAF(j)))$. $\#TAF$ denotes the number of distinct $TAF(i)$ $(1 \leq i \leq n)$ $\$TAF$ stands for the number of one-to-one correspondences of A and B and is defined as

$$\$TAF = \sum_{i=1}^{n} |TAF(i)|$$

Different from the association function of OCS defined in [22], a triangulate association function is a full mapping between two sequences ($A \to B$), as shown in Figure 7, which shows $TAF(1) = TAF(2) = TAF(3) = [1,1]$, $TAF(4) = [2,4]$ and $TAF(5) = [5,5]$. $|TAF(1)| = 1$ and $|TAF(4)| = 3$. $\#TAF = 7$ and $\$TAF = 15$. a_i's correspondence is $b_{TAF(i)}$. If TAF$(i) = [x,y]$, then $b_{TAF(i)} = b_{[x,y]} = b_xb_{x+1}, \ldots, b_y$. $TAF_{A \to B}$ has a corresponding $TAF_{B \to A}$. Figure 7 also shows a full mapping ($B \to A$) such that $TAF(1) = [1,3]$ and $TAF(2) = [4,4]$. Notice that $\#TAF_{A \to B} = \#TAF_{B \to A}$ and $\$TAF_{A \to B} = \$TAF_{B \to A}$. For simplicity, $TAF_{A \to B}$ and $TAF_{B \to A}$ are used to represent a single TAF hereinafter.

Definition 4.2. *Optimal triangulate correspondence of sequences without merging* (OTCS/o). Given two sequences $A = a_1a_2a_3, \ldots, a_n$ and $B = b_1b_2b_3, \ldots, b_m$, the OTCS/o problem

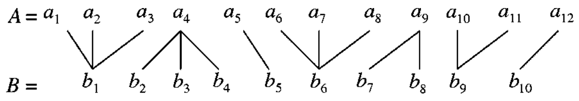

$A = a_1$ a_2 a_3 a_4 a_5 a_6 a_7 a_8 a_9 a_{10} a_{11} a_{12}
$B =$ b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8 b_9 b_{10}

Figure 7. A triangulate association mapping between two sequences

of A and B is to find a TAF of A and B such that the total penalty

$$FMP_{TAF}^{OTCS/o}(A, B) = \sum_{\forall i(j \in TAF(i))} p(a_i, b_j)$$

is minimized. p is a penalty function.

Based on OTCS/o, the distance between A and B is defined as follows:

$$D_{full}^{OTCS/o}(A, B) = \frac{\min\limits_{\forall TAF} FMP_{TAF}^{OTCS/o}(A, B)}{\$TAF} \tag{10}$$

where the division by $\$TAF$ is an averaging process which is omitted by conventional string-matching algorithms [6, 24]. This avoids matching long sequences which accumulates errors. Equation (10) can be evaluated by a dynamic programming procedure to determine the minimum distance path through a penalty table, as shown in Algorithm 4.1, which produces a triangulate association mapping (see Lemma 4.1 and Theorem 4.2 in Appendex A).

Algorithm 4.1. Distance_OTCS/o$(A = a_1a_2a_3, \dots, a_n,\ B = b_1b_2b_3, \dots, b_m, p)$

```
$D(0,0) = 0$; $N(0,0) = 0$;
for $i = 1$ to $n$ do begin $D(i,0) = \infty$; $N(i,0) = 0$; end;
for $j = 1$ to $m$ do begin $D(0,j) = \infty$; $N(0,j) = 0$; end;
  for $i = 1$ to $n$ do
    for $j = 1$ to $m$ do begin
    $t_1 = D(i-1, j-1)$; $t_2 = D(i-1, j)$; $t_3 = D(i, j-1)$;
    $D(i,j) = t_1 + p(a_i, b_j)$; $N(i,j) = N(i-1, j-1) + 1$;
    if ($t_2 < t_1$ and $t_2 < t_3$) then begin $D(i,j) = t_2 + p(a_i, b_j)$; $N(i,j) = N(i-1,j) + 1$; end;
    if ($t_3 < t_1$ and $t_3 < t_2$) then begin $D(i,j) = t_3 + p(a_i, b_j)$; $N(i,j) = N(i,j-1) + 1$; end;
  end;
return $D(n, m)/N(n, m)$;
```

The penalty table $D(i,j)$ determines an optimal matching of two sequences and the correspondence can be traced back to construct the desired TAF. The complexity of Algorithm 4.1 is $O(nm)$ $(= O(n^2) \quad if\ n = m)$, where p is evaluated by a lookup table in constant time.

Consider the following sequences $A = 0670$, $B = 01070$, $C = 02070$ and $D = 00067700$, where each symbol/digit (from 0 to 7) represents a moving direction (from East to Southeast anti-clockwise) on a plane, as shown in Figure 8. Some researches [12,13,16] use a series of chain codes to specify a sequence of eight possible directions of the trajectory. Suppose the distance between two symbols a and b is $\min(|a-b|, 8-|a-b|)$. According to OTCS/o, $D(A, B) = 0.33$, $D(A, C) = 0.5$ and $D(A, D) = 0$. We have $TAF_{A\to B} = TAF_{A\to C} = \{(1, [1,3]),\ (2,[4,4]),(3, [4,4]),\ (4, [5,5])\}$. If we adopt the OCS algorithm whose miss penalty is set to be 2, then $D(A, B) = D(A, C) = 4$, and $D(A, D) = 8$. We have the association mapping (from B or C to A) F: $\{1,2,3,4,5\} = \{1, \perp, 2,3,4\}$ satisfying OCS, where $\perp$ is a null symbol denoting the second line segments of B and C match nothing. Compared with OCS in

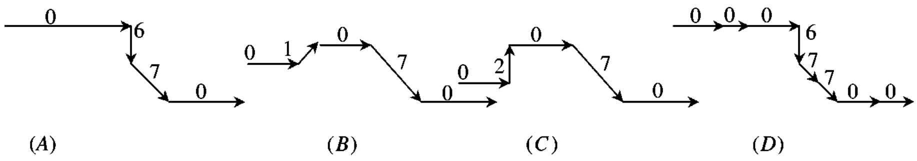


Figure 8. An example of trajectory representation

this example, OTCS/o distinguishes the detailed differences between AB and AC more efficiently than OCS, which has limitation on discriminating miss penalties. Moreover, from the viewpoint of direction, A and D are the same obviously. D may be produced by the proposed model introduced in Section 3 since two adjacent motion/relation clips may have identical directions but different colors or sizes. OCS fails to detect the equivalence between A and D.

Definition 4.3. *Optimal triangulate correspondence of sequences with merging* (OTCS/w). Given two sequences $A = a_1a_2a_3, \ldots, a_n$ and $B = b_1b_2b_3, \ldots, b_m$, the OTCS/w problem of A and B is to find a TAF of A and B such that the total penalty

$$FMP_{TAF}^{OTCS/w}(A, B) = \sum_{\forall i(|TAF_{A\to B}(a_i)| \geq 1)} p(a_i, TAF_{A\to B}(a_i)) + \sum_{\forall j(|TAF_{B\to A}(b_j)| > 1)} p(b_j, TAF_{B\to A}(b_j))$$

is minimized, where p is a penalty function evaluating the dissimilarity between one symbol and a subsequence.

Based on OTCS/w, the distance between A and B is defined as follows:

$$D_{full}^{OTCS/w}(A, B) = \frac{\min_{\forall TAF} FMP_{TAF}^{OTCS/w}(A, B)}{\# TAF} \tag{11}$$

Algorithm 4.2. Distance_OTCS/w($A = a_1a_2a_3, \ldots, a_n$, $B = b_1b_2b_3, \ldots, b_m, p$)

```
$D(0,0) = 0$; $N(0,0) = 0$;
for $i = 1$ to $n$ do begin $D(i,0) = \infty$; $N(i,0) = 0$; end;
for $j = 1$ to $m$ do begin $D(0,j) = \infty$; $N(0,j) = 0$; end;
for $i = 1$ to $n$ do
  for $j = 1$ to $m$ do begin
    min_value = $t_{i-1,j-1} = D(i-1, j-1) + p(a_i, b_j)$;
    for x = 1 to $i$-2 do $t_{x,j-1} = D(x, j-1) + p(b_j, a_{x+1} \ldots a_i)$;
    for y = 1 to $j$-2 do $t_{i-1,y} = D(i-1, y) + p(a_i, b_{y+1} \ldots b_j)$;
    for each $t_{u,v}$(($u = 1..i$-2, $v = j-1$) and ($u = i-1$, $v = 1..j$-2))
    if ($t_{u,v} <$ min_value) then begin $D(i,j) = t_{u,v}$; $N(i,j) = N(u,v) + 1$; min_value = $t_{u,v}$; end;
  end;
return $D(n,m)/N(n,m)$;
```

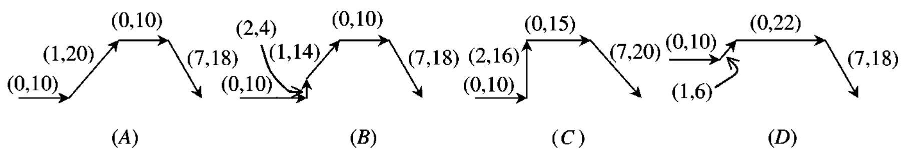


Figure 9. Trajectory representation in the example

This algorithm produces the optimal matching (see Theorem 4.1 in Appendix A) with the complexity of $O(nm(n+m))$ ($= O(n^3)$ *if* $n = m$), where p can be evaluated by a lookup table in constant time. The desired TAF can be generated by tracking back the correspondence in the penalty table $D(i, j)$. Furthermore, OTCS/o is a special case of OTCS/w (see Lemma 4.2 in Appendix A).

As far as trajectory matching is concerned, OTCS/o works only with the direction feature; however, OTCS/w works with both direction and displacement features. For example, Figure 9 illustrates three trajectories of $A = (0,10)(1,20)(0,10)(7,18)$, $B = (0,10)(2,4)(1,14)(0,10)(7,18)$, $C = (0,10)(2,16)(0,15)(7,20)$ and $D = (0,10)(1,6)(0,22)(7,18)$, where each symbol is represented by the pair of (direction, displacement/length). According to OTCS/o, we have $TAF_{A \to B} = \{(1,[1,1]),(2,[2,3]),(3,[4,4]),(4,[5,5])\}$, $D(A, B) = D(A, C) = 1$, $D(A, D) = 0$. If the OTCS/w algorithm is taken, in which the distance between a symbol a and a subsequence $b_1, \ldots, b_n$ is $|a\text{'s len} - \sum_{\forall i} b_i\text{'s len}| + 5 \cdot \sum_{\forall i} \min(|a\text{'s dir} - b_i\text{'s dir}|, 8 - |a\text{'s dir} - b_i\text{'s dir}|)$, we have $D(A, B) = 7$, $D(A, C) = 16$ and $D(A, D) = 26$. Compared with OTCS/o, OTCS/w can merge adjacent symbols such that the total displacement is considered in the similarity measure.

4.2. Partial Matching

Full matching can be changed into partial matching by attaching dummy symbols to the beginning and end of sequences. Given two sequences $A = a_1 a_2 a_3, \cdots, a_n$ and $B = b_1 b_2 b_3, \ldots, b_m$, two dummy symbols a_d and b_d are introduced. Let $A' = a_d a_1 a_2 a_3, \ldots, a_n a_d$ and $B' = b_d b_1 b_2 b_3, \ldots, b_m b_d$. Partial matching of A and B can be converted into full matching of A' and B', as shown in Figure 10. $p(a_d, b_x, \ldots, b_y)$ and $p(b_d, a_x, \ldots, a_y)$ are so-called miss penalties. $p(a_d, b_d)$ is set to be 0 in default.

Assume that the situation in Figure 10 is a triangulate mapping between A' and B' such that the total penalty is minimized. Let the matched subsequences be $A'' = a_3 a_4, \ldots, a_8$ and $B'' = b_1 b_2, \ldots, b_7$. For the partial matching of OTCS/o, the total distance is computed as

$$D_{partial}^{OTCS/o}(A', B') = D_{full}^{OTCS/o}(A'', B'') + \sum_{\forall a_d} p(a_d, b_j) + \sum_{\forall b_d} p(b_d, a_i) \tag{12}$$

For the partial matching of OTCS/w, the total distance is computed as

$$D_{partial}^{OTCS/w}(A', B') = D_{full}^{OTCS/w}(A'', B'') + \sum_{\forall a_d} p(a_d, TAF_{A \to B}(a_d))$$
$$+ \sum_{\forall b_d} p(b_d, TAF_{B \to A}(b_d)) \tag{13}$$

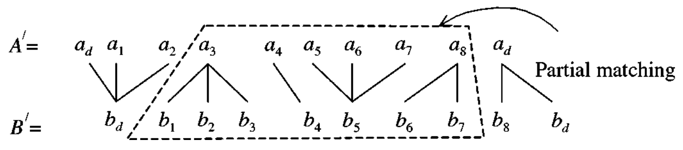

$A' =$
a_d a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8 a_d
Partial matching
$B' =$
b_d b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8 b_d

Figure 10. A partial matching of two sequences A and B

The more mismatched the symbols, the larger is the total distance. Generally, mismatched symbols are independent of each other. $p(a_d, b_x, \ldots, b_y) = \sum_{j=x,\ldots,y} p(a_d, b_j)$ and $p(b_d, a_x, \ldots, a_y) = \sum_{i=x,\ldots,y} p(a_i, b_d)$.

Generally, a user can insert dummy symbols into a query sequence arbitrarily. The algorithms which evaluate Eqs. (12) and (13) are similar to Algorithms 4.1 and 4.2 and have the complexity of $O((n+k)(m+k))$ $(=O(n^2)$ *if* $n=m)$ and $O((n+k)(m+k)(n+m+k))$ $(=O(n^3)$ *if* $n=m)$, respectively, where k is the number of dummy symbols.

5. Performance Evaluation

This section presents the results of experiments that evaluate the performance of the proposed matching algorithms from Section 4. The performance metric, test database and simulation parameters are described and, then, the experimental results and related analysis are given.

5.1. Performance Metric

There is no standard to decide the degree of similarity in all applications, particularly because the trajectory-matching problem generally depends on users' perceptions and requirements. Current research on curve and shape recognition determines the similarity between two given trajectories by the following four policies, as illustrated in Figure 11. Two methods compute distance based on the distance each vertex of a trajectory projects onto the other one while the others are based on the distance between corresponding equidistant points on the two trajectories. None of the four policies are dependent on the computation of distances. This work uses Euclidean distance (the *dist* function) to make a meaningful comparison.

Suppose a trajectory T is a concatenation of individual line segment chains represented by an ordered list of N vertices $T = \{u_i | u_i = (x_i, y_i), 1 \le i \le N\} = \{S_j | S_j = \overline{u_j u_{j+1}}, 1 \le j \le N-1\}$, where (x_i, y_i) is the $x-y$ coordinates of a vertex. If a point u lies on S_j, the path length between u_1 and u is $\sum_{k=1,\ldots,j-1} |S_k| + |u_j u|$, denoted by $P_T(u_1, u)$. For any two points u_i and u_j on T, their path length $P_T(u_i, u_j)$ is $|P_T(u_1, u_i) - P_T(u_1, u_j)|$. An arbitrary fragment of a trajectory T from point u_i to point u_j is denoted as $T[u_i, u_j]$, where $P_T(u_1, u_j) \ge P_T(u_1, u_i)$. Details of similarity measure between two trajectories $T_1 = \{t_1, t_2, \ldots, t_N\}$ and $T_2 = \{u_1, u_2, \ldots, u_M\}$, as illustrated in Figure 11, are defined as follows:

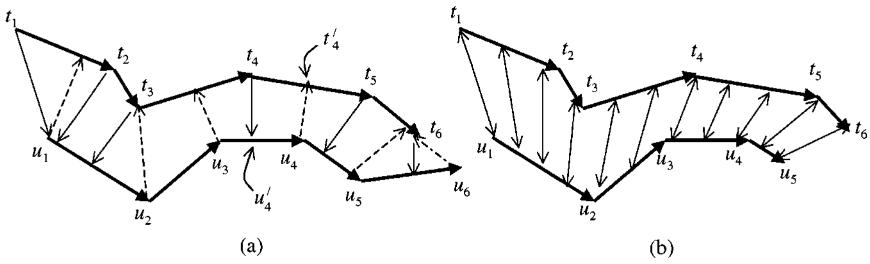


Figure 11. Four trajectory-matching policies: **(a)** average/maximum distance bases on vertices' projections; **(b)** average/maximum distance based on equidistant points

(1) The average distance based on vertices' projections (AP) is defined as

$$AP(T_1, T_2) = \frac{\sum_{i=1}^{N} dist(t_i, u'_i) + \sum_{j=1}^{M} dist(u_j, t'_j)}{N + M}$$

where $u'_i = Proj(t_i, T_2[u'_{i-1}, u_M])$, $t'_j = Proj(u_j, T_1[t'_{j-1}, t_N])$, $Proj(t_i, T_2[u'_{i-1}, u_M])$ and $Proj(u_j, T_1[t'_{j-1}, t_N])$, respectively, return t_i's projection point on the T_2's fragment from u'_{i-1} to u_M and u_j's projection point on the T_1's fragment from t'_{j-1} to t_N, $u'_0 = u_1$, and $t'_0 = t_1$. In our experiment, we restrict the projection range to satisfy the directional property of trajectories, which shape recognition lacks.

(2) The maximum distance based on vertices' projections (MP) is defined as

$$MP(T_1, T_2) = \mathrm{Max}\left(\bigcup_{i=1}^{N} dist(t_i, u'_i), \bigcup_{j=1}^{M} dist(u_j, t'_j) \right)$$

where u'_i and t'_j are defined in AP.

(3) The average distance based on equidistant points (AE) is defined as

$$AE(T_1, T_2, n) = \frac{\sum_{i=1}^{n} dist(t''_i, u''_i)}{n}$$

where $t''_1 = t_1$, $u''_1 = u_1$, $t''_n = t_N$, $u''_n = u_M$, and $P_{T_1}(t''_i, t''_{i+1}) = P_{T_1}(t_1, t_N)/(n-1)$ and $P_{T_2}(u''_i, u''_{i+1}) = P_{T_2}(u_1, u_M)/(n-1)$ for $i = 1,..,n-1$. T_1 and T_2 are divided into $n-1$ fragments with equal length by n points.

(4) The maximum distance based on equidistant points (ME) is defined as

$$AE(T_1, T_2, n) = \mathrm{Max}\left(\bigcup_{i=1}^{n} dist(t''_i, u''_i) \right)$$

where t''_i and u''_i are defined in AE.

These four trajectory-matching policies simulating user-defined preference relation among trajectories are designed to be the basis for judging the retrieval performance of motion-matching algorithms. To examine the impact of the proposed matching algorithms on motion recognition, the study experiments with optimal correspondence of subsequences (OCS) [24] (conventional string-matching algorithm), the VIOLONE approach [16], optimal triangulate correspondence of sequences without merging (OTCS/o) and optimal triangulate correspondence of sequences with merging (OTCS/w). According to the four trajectory-matching policies, the penalty function in Definition 4.3 can be evaluated by AP, MP, AE, or ME. Restated, OTCS/w can be categorized into OTCS/w + AP, OTCS/w + MP, OTCS/w + AE, or OTCS/w + ME depending on the penalty function design.

Suppose the rank ordering of database trajectories induced by the investigated algorithms (Δ^{system}) are F_{OCS}, $F_{VIOLONE}$, $F_{OTCS/o}$, $F_{OTCS/w+AP}$, $F_{OTCS/w+MP}$, $F_{OTCS/w+AE}$ and $F_{OTCS/w+ME}$. The rank ordering of trajectories induced by the four basis policies (Δ^{expert}) are F_{AP}, F_{MP}, F_{AE} and F_{ME}. Three main performance metrics used are recall (R_k), precision (P_k) and R_{norm} [25], where k is a threshold varying between 1 and database size. R_k and P_k are defined as

$$R_k(\Delta^{system}, \Delta^{expert}) = \frac{|\{\text{the top } k \text{ objs in } \Delta^{system}\}_{\text{¡}}\text{ä}\{\text{the top } k \text{ objs in } \Delta^{expert}\}|}{|\{\text{the top } k \text{ objs in } \Delta^{expert}\}|}$$

and

$$P_k(\Delta^{system}, \Delta^{expert}) = \frac{|\{\text{the top } k \text{ objs in } \Delta^{system}\}_{\text{¡}}\text{ä}\{\text{the top } k \text{ objs in } \Delta^{expert}\}|}{\{\text{the top } k \text{ objs in } \Delta^{system}\}}$$

A straightforward way to evaluate the permutation quality of results is to use R_{norm} [23]. As users prefer retrieving the top k relevant trajectories instead of all of the trajectories, the domain examined in this study only includes the top k relevant results. Then R^k_{norm} is defined as

$$R^k_{norm}(\Delta^{system}, \Delta^{expert}) = R_{norm}(\Delta^{system}_k, \Delta^{expert}_k) = \frac{1}{2}\left(1 + \frac{S^+ - S^-}{S^+_{Max}}\right)$$

where

$\Delta^{system}_k = \{$the top k objs in $\Delta^{system}\} - (\{$the top k objs in $\Delta^{system}\} - \{$the top k objs in $\Delta^{expert}\})$ and $\Delta^{expert}_k = \{$the top k objs in $\Delta^{expert}\} - (\{$the top k objs in $\Delta^{expert}\} - \{$the top k objs in $\Delta^{system}\})$.

S^+ is the number of trajectory pairs where a better trajectory is ranked ahead of a worse one by Δ^{system}_k, S^- the number of trajectory pairs where a worse trajectory is ranked ahead of a better one by Δ^{system}_k, and S^+_{Max} the maximum possible number of S^+ from Δ^{exoert}_k. R_{norm} values vary from 0 to 1. The larger the R_{norm} value, the closer the rank ordering.

5.2. Test Database and Simulation Parameters

This study generates a test database with 120 trajectories by rotating 15 different S-shape samples, from 0, $\pi/8$ to $7\pi/8$. Each trajectory is normalized in advance as explained in Section 3.3 and represented as a set of directions and displacements. A pattern selected from the database randomly is considered to be the query trajectory. An exhaustive searching algorithm has been implemented to minimize the proximity measurements for all possible relative positions of the two trajectories. The basis policies (AP, MP, AE and ME) are independent of the scale and the translation of trajectories. The algorithms investigated herein are OCS, VIOLONE, OTCS/o, OTCS/w + AP, OTCS/w + MP, OCTS/w + AE and OCTS/w + ME. As Section 4 explained, OCS, VIOLONE and OTCS/o merely consider the directional feature of the trajectory, which is independent of the scale and the translation of the trajectories. Meanwhile, OTCS/w can satisfy the conditions of scale and translation by normalizing each displacement and shifting one of the trajectories to the center of gravity of the other.

To verify the merits of OTCS/w, this study conducted two experiments to compare its performance with OCS and OTCS/o under various matching situations, including full matching and partial matching. The difference between two directions $\vec{x}$ and $\vec{y}$ is computed by $15 \times (1 - (\vec{x} \cdot \vec{y}) / |\vec{x}||\vec{y}|)$, which varies from 0 to 30. The top 30 ranked trajectories are taken as the results which users expect (Δ^{expert}) for each experiment.

5.3. Experimental Results

In the experiment of full matching, we slightly modify OCS such that the first and last symbols must be matched and miss penalty is set to be 10. The number of retrieved objects k from Δ^{system} varies from 10 to 60. Figures 12–15 show the recall (R_k) and precision (P_k) results based on the criteria of four full trajectory-matching policies.

In the experiment of partial matching, each symbol's miss penalty is set to be 10 for all methods. The number of retrieved objects k from Δ^{system} varies from 10 to 60. Figures 16–19 show the recall (R_k) and precision (P_k) results based on the criteria of four partial trajectory-matching policies.

Experimental results indicate that at either full matching or partial matching, the order of performance in maximizing recall and precision is OTCS/w > OTCS/o > VIOLONE > OCS. Namely, OTCS/w performs best and OCS performs worst for the

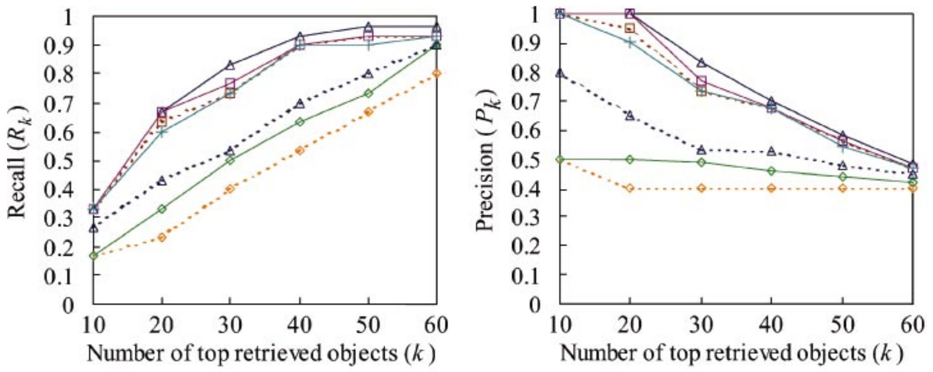


Figure 12. *Recall* and *Precision* under the criterion *AP* (full matching): (- ○ -), OCS; (—◇—), VIOLONE; (- △ -), OTCS/o; (—▲—), OTCS/w-AP; (- □ -), OTCS/w-MP; (—⊟—), OTCS/w-AE; (—+—), OTCS/w-ME

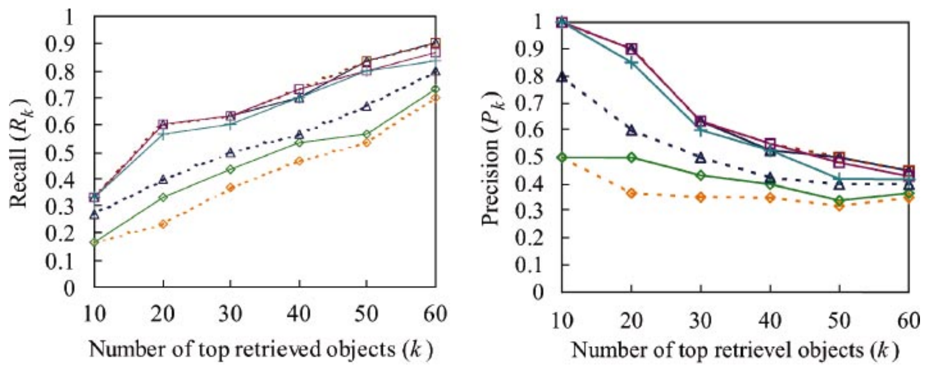


Figure 13. *Recall* and *Precision* under the criterion *MP* (full matching): (- ○ -), OXΣ; (—◇—), ζ IOΛONE; (- △ -), OTXΣ/o; (—▲—), OTXΣ/ω-AΠ; (- □ -), OTXΣ/ω-M; (—⊟—), OTXΣ/ω-AE; (—+—), OTXΣ/ω-ME

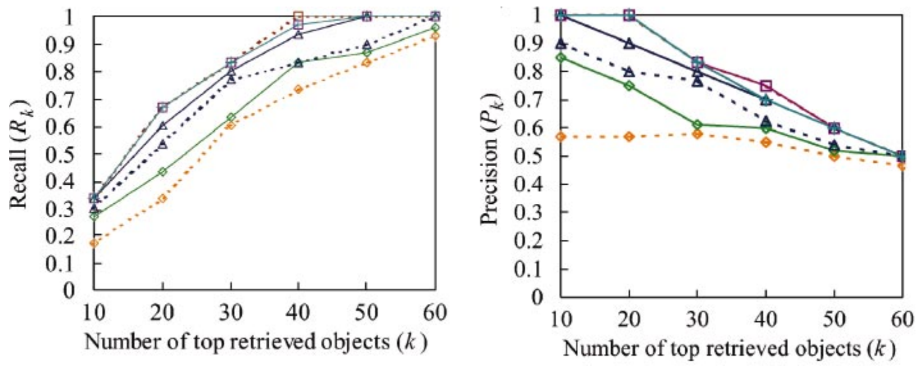


Figure 14. *Recall* and *Precision* under the criterion *AE* (full matching): (- ○ -), OCS; (—◇—), VIOLONE; (- △ -), OTCS/o; (—▲—), OTCS/w-AP; (- □ -), OTCS/w-MP; (—⊟—), OTCS/w-AE; (—+—), OTCS/w-ME

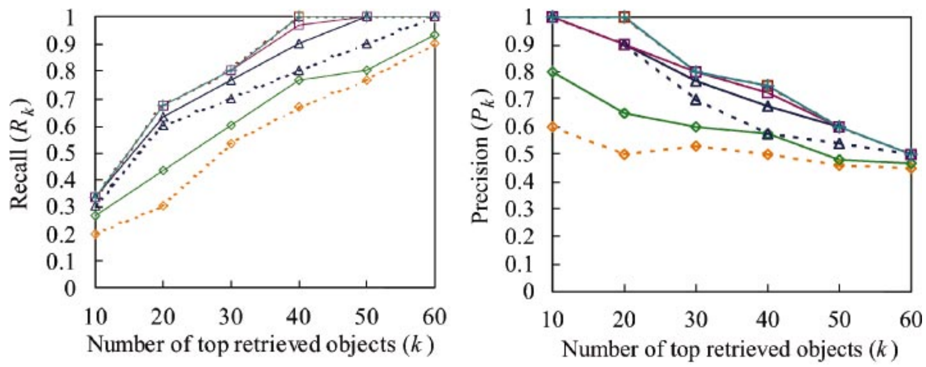


Figure 15. *Recall* and *Precision* under the criterion *ME* (full matching): (- ○ -), OCS; (—◇—), VIOLONE; (- △ -), OTCS/o; (—▲—), OTCS/w-AP; (- □ -), OTCS/w-MP; (—⊟—), OTCS/w-AE; (—+—), OTCS/w-ME

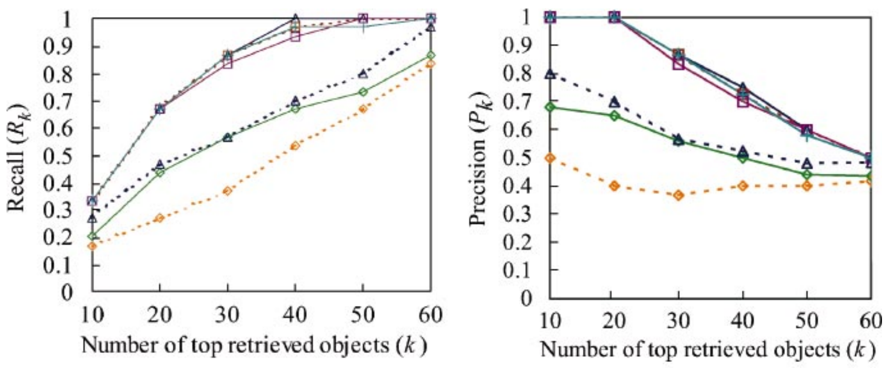


Figure 16. *Recall* and *Precision* under the criterion *AP* (partial matching): (- ○ -), ΟΧΣ; (—◇—), ζ ΙΟΛΟΝΕ; (- △ -), ΟΤΧΣ/ο; (—▲—), ΟΤΧΣ/ω-Α*Π*; (- □ -), ΟΤΧΣ/ω-Μ; (—⊟—), ΟΤΧΣ/ω-ΑΕ; (—+—), ΟΤΧΣ/ω-ΜΕ

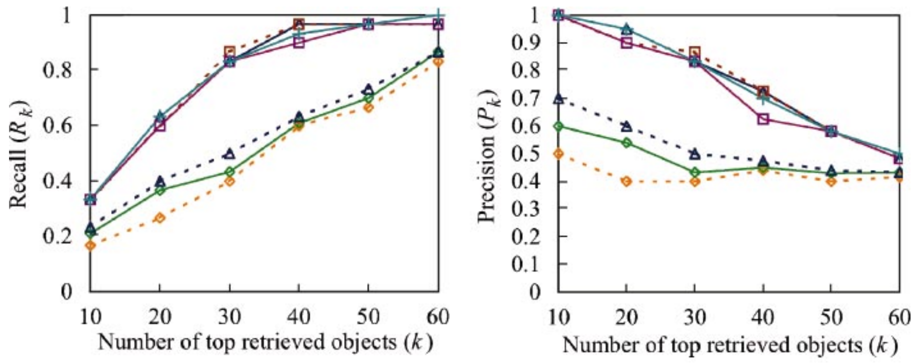


Figure 17. *Recall* and *Precision* under the criterion *MP* (partial matching): (- ○ -), OCS; (—◇—), VIOLONE; (- △ -), OTCS/o; (—▲—), OTCS/w-AP; (- □ -), OTCS?w-M

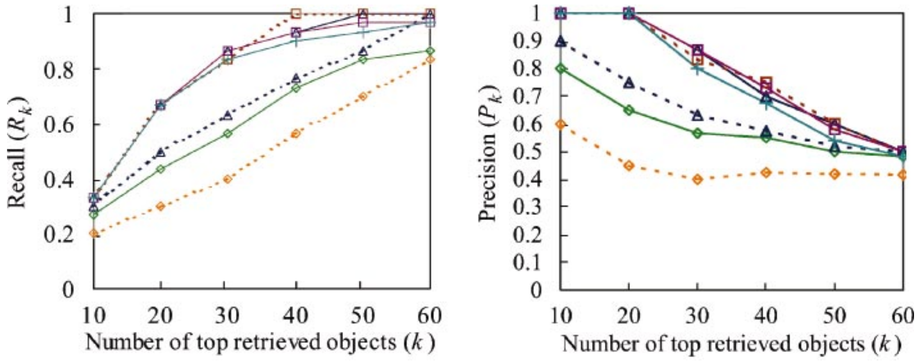


Figure 18. *Recall* and *Precision* under the criterion *AE* (partial matching): (- ○ -), OCS; (—◇—), VIOLONE; (- △ -), OTCS/o; (—▲—), OTCS/w-AP; (- □ -), OTCS/w-MP; (—⊟—), OTCS/w-AE; (—+—), OTCS/w-ME

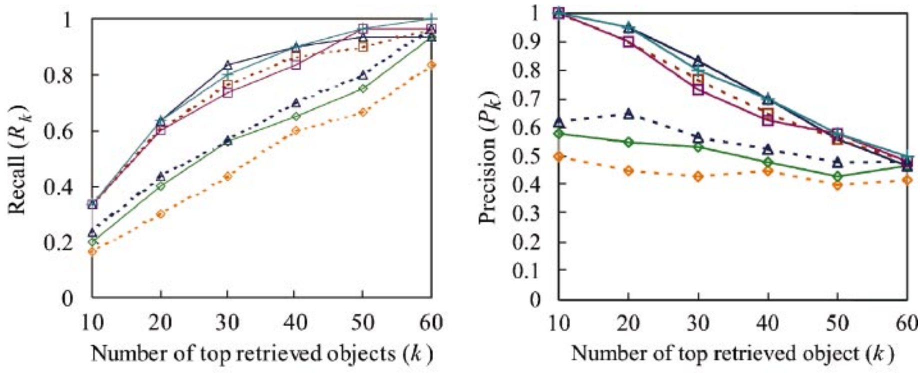


Figure 19. *Recall* and *Precision* under the criterion *ME* (partial matching): (- ○ -), OCS; (—◇—), VIOLONE; (- △ -), OTCS/o; (—▲—), OTCS/w-AP; (- □ -), OTCS/w-MP; (—⊟—), OTCS/w-AE; (—+—), OTCS/w-ME

Table 1. The experimental result of R^k_{norm}

	Match							
	Full matching				Partial matching			
Thre-shold	Method							
$k =$	OCS	VIOLONE	OTCS/o	OTCS/w	OCS	VIOLONE	OTCS/o	OTCS/w
30	0.626	0.652	0.754	0.846	0.462	0.565	0.711	0.846
60	0.671	0.671	0.764	0.837	0.387	0.522	0.740	0.850
90	0.689	0.692	0.762	0.834	0.421	0.543	0.750	0.849
120	0.696	0.693	0.763	0.834	0.421	0.544	0.750	0.851

four criteria of AP, MP, AE and ME. The main reason for this result is that OTCS/w incorporates direction and displacement features, while displacement is absent from the others. OTCS/o performs better than VIOLONE because OTCS/o allows one-to-many and many-to-one matching while VIOLONE merely allows one-to-many matching between a query object and a database object. VIOLONE performs better than OCS because OCS omits the characteristic of continuation of a trajectory. When the penalty function of OTCS/w is consistent with the criterion, it generally performs better. For example, OTCS/w-AP performs best for the criterion AP. However, the performance difference among all OTCS/w's is smaller than that among OCS, VIOLONE, OTCS/o and OTCS/w. Thus, algorithm selection appears more important than the design of penalty functions.

The rank ordering of extracted trajectories can be compared with that of the trajectories which users expect by observing the R^k_{norm}, as shown in Table 1. Experimental results indicate that at either full matching or partial matching, the order of performance in maximizing R^k_{norm} is OTCS/w > OTCS/o > VIOLONE > OCS. In other words, OTCS/w provides better permutation quality of the retrieved results than the others. As the order of computational complexity is OTCS/w > OTCS/o = OCS > VIOLONE, our system takes OTCS/w as the query-processing algorithm when both features, direction and displacement, are considered while OTCS/o is taken when only direction feature is considered. In addition, determining the missed penalty of OCS when there is full matching is inconvenient and difficult for a less experienced user while no parameters are required by the others.

6. Implementation

This section presents the overall architecture of a prototype system, which was developed based on the proposed visual aggregation model in the Microsoft Window environment.

The system has two main modules: *storage* and *retrieval modules*, as shown in Figure 20. In the storage module, a video sequence is decomposed into scenes using the histogram-based technology [19] and then moving objects in a scene are tracked using the template-based technology [9]. The extracted moving objects are represented as motion sequences. For each motion sequence, a motion composition is automatically created and normalized. With the assistance of composition operations, as illustrated in Figure 21, users can recursively group the generated motion compositions in the database into a complex one, whose features are analyzed and normalized if necessary. If two arbitrary motion sequences in a motion composition overlap, the system automatically generates a corresponding relation sequence. In the retrieval module, the system accepts a request specified by drawing example motions, extracts and normalizes the appropriate visual features and, then, performs motion-matching algorithms, as shown in Figure 22.

The proposed visual aggregation model has been integrated into relational databases in our system, which maintains 13 main tables, as shown in Figure 23. The *RVT table* maintains bibliographic data of video in the database. The SDT table describes scenes. The FST, RST and MST tables present feature sequences, relation sequences and motion sequences, respectively, while the FCT, RCT and MCT tables describe the

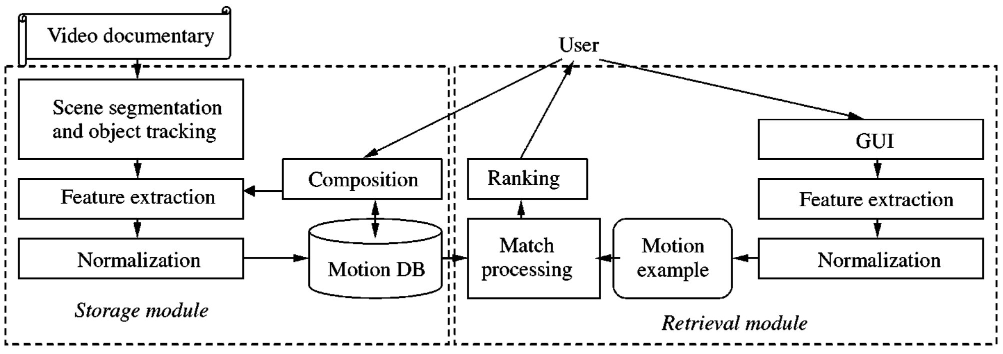
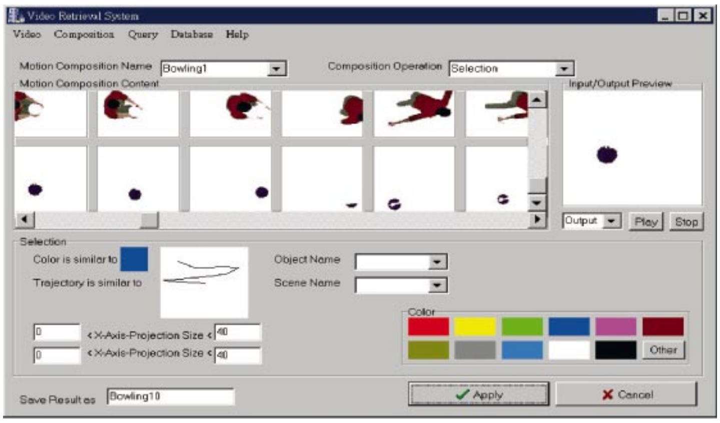
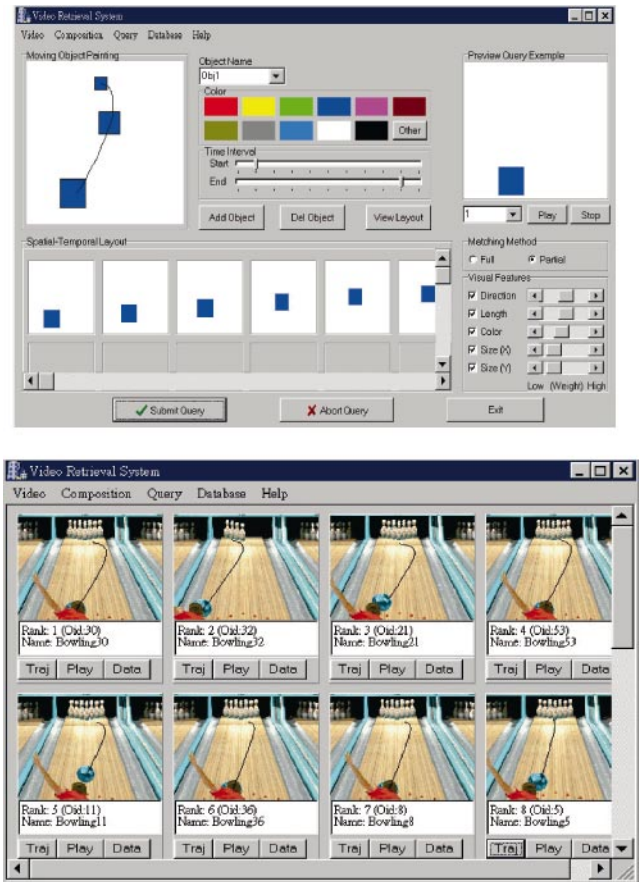


Figure 20. The architecture of the prototype system



Motion Composition Name: Bowling1
Composition Operation: Selection
Motion Composition Content
Color is similar to
Trajectory is similar to
Object Name
Scene Name
0 < X-Axis-Projection Size < 40
0 < X-Axis-Projection Size < 40
Save Result as: Bowling10

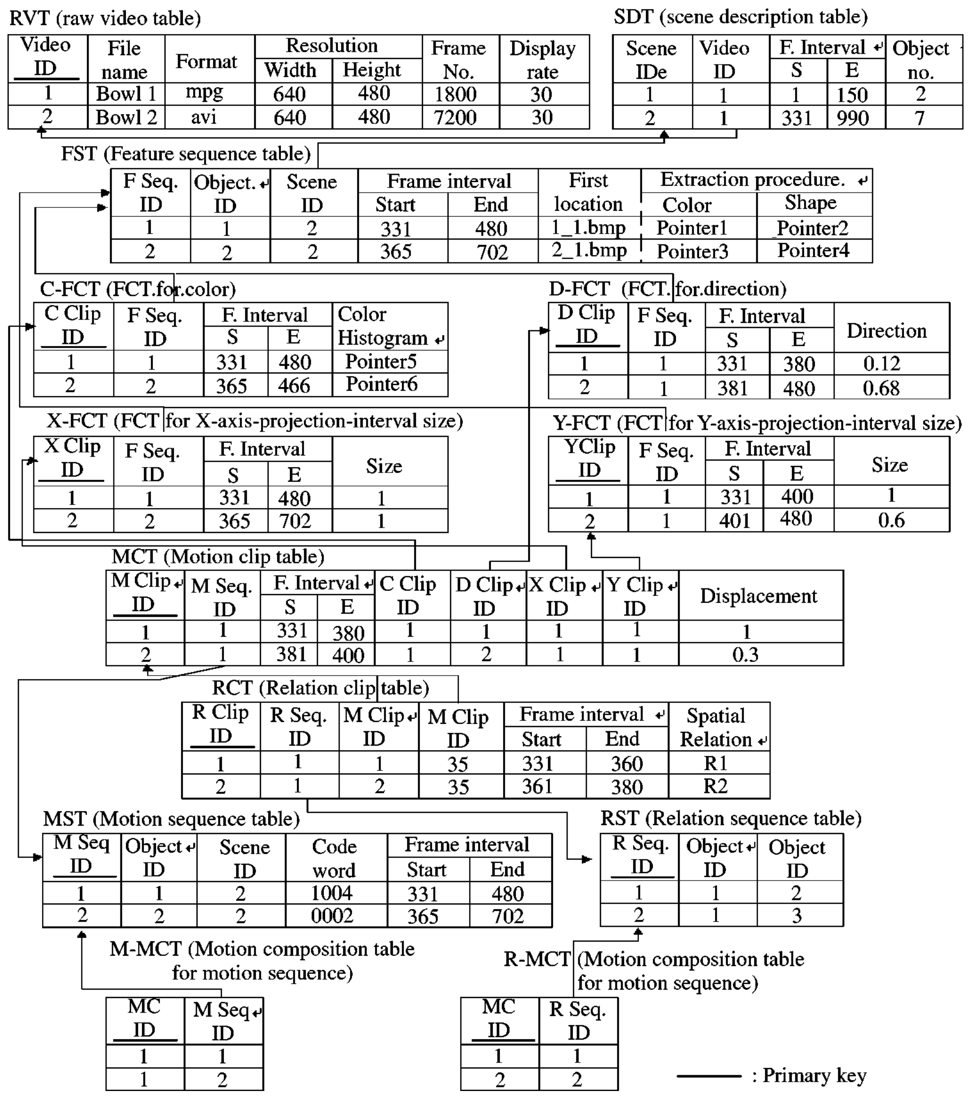
Figure 21. The interfaces of *select* operation



Object Name: Obj1
Matching Method: ○ Full ◉ Partial
☑ Direction
☑ Length
☑ Color
☑ Size (X)
☑ Size (Y)

Rank: 1 (Oid:30) Name: Bowling30
Rank: 2 (Oid:32) Name: Bowling32
Rank: 3 (Oid:21) Name: Bowling21
Rank: 4 (Oid:53) Name: Bowling53
Rank: 5 (Oid:11) Name: Bowling11
Rank: 6 (Oid:36) Name: Bowling36
Rank: 7 (Oid:8) Name: Bowling8
Rank: 8 (Oid:5) Name: Bowling5

Figure 22. An example query for the bowling application



RVT (raw video table)

Video ID	File name	Format	Resolution		Frame No.	Display rate
			Width	Height		
1	Bowl 1	mpg	640	480	1800	30
2	Bowl 2	avi	640	480	7200	30

SDT (scene description table)

Scene IDe	Video ID	F. Interval		Object no.
		S	E	
1	1	1	150	2
2	1	331	990	7

FST (Feature sequence table)

F Seq. ID	Object. ID	Scene ID	Frame interval		First location	Extraction procedure.	
			Start	End		Color	Shape
1	1	2	331	480	1_1.bmp	Pointer1	Pointer2
2	2	2	365	702	2_1.bmp	Pointer3	Pointer4

C-FCT (FCT.for.color)

C Clip ID	F Seq. ID	F. Interval		Color Histogram
		S	E	
1	1	331	480	Pointer5
2	2	365	466	Pointer6

D-FCT (FCT.for.direction)

D Clip ID	F Seq. ID	F. Interval		Direction
		S	E	
1	1	331	380	0.12
2	1	381	480	0.68

X-FCT (FCT for X-axis-projection-interval size)

X Clip ID	F Seq. ID	F. Interval		Size
		S	E	
1	1	331	480	1
2	2	365	702	1

Y-FCT (FCT for Y-axis-projection-interval size)

YClip ID	F Seq. ID	F. Interval		Size
		S	E	
1	1	331	400	1
2	1	401	480	0.6

MCT (Motion clip table)

M Clip ID	M Seq. ID	F. Interval		C Clip ID	D Clip ID	X Clip ID	Y Clip ID	Displacement
		S	E					
1	1	331	380	1	1	1	1	1
2	1	381	400	1	2	1	1	0.3

RCT (Relation clip table)

R Clip ID	R Seq. ID	M Clip ID	M Clip ID	Frame interval		Spatial Relation
				Start	End	
1	1	1	35	331	360	R1
2	1	2	35	361	380	R2

MST (Motion sequence table)

M Seq ID	Object ID	Scene ID	Code word	Frame interval	
				Start	End
1	1	2	1004	331	480
2	2	2	0002	365	702

RST (Relation sequence table)

R Seq. ID	Object ID	Object ID
1	1	2
2	1	3

M-MCT (Motion composition table for motion sequence)

MC ID	M Seq ID
1	1
1	2

R-MCT (Motion composition table for motion sequence)

MC ID	R Seq. ID
1	1
2	2

Figure 23. The storage structure of the system

corresponding clips. Motion composition table comprises the M-MCT and R-MCT tables. For selection operations, indexing on the color attribute of the C-FCT table can facilitate the nearest-neighbor queries in a multidimensional space such as R-tree. The direction and size attributes of the D-FCT, X-FCT and Y-FCT tables can be indexed by B-tree and hashing. For projection operations, a two-dimensional indexing structure is used to evaluate temporal range queries. The system also provides a filtering mechanism [8], which preprocesses the database objects before querying processing.

7. Conclusions

In this paper, we presented a visual aggregation model for the representation of visual features (trajectory, color and size) of a single moving object and the directional and topological relations among multiple objects in video databases. A set of composition operations is introduced for the purpose of calculating object motions at various levels of semantic granularity. Two algorithms of triangulate matching with/without merging are designed for motion recognition, which can be applied into other applications such as shape, curve and handwritten recognition. Experimental results indicate that at either full matching or partial matching, the proposed algorithms significantly improve trajectory-matching performance of the conventional approaches.

Some issues need to be addressed about querying by motion examples in video databases. In some applications, people usually focus a camera on a moving object of interest to them. Therefore, the object is always located in the center of the viewport and its trajectory would be hidden from the camera operation. We are investigating how to evaluate object motion from the movement of its background and compute the corresponding coordinate sequence of a moving object. To automatically determine the thresholds mentioned in the processing of normalization in our system, we are planning to apply image-segmentation technologies such as local gradient and Laplacian computations. In addition to sports event analysis such as bowling applications, we also intend to extend our system to support other applications, including surveillance and satellite image sequences.

Acknowledgment

The authors would like to thank the National Science Council of the Republic of China for financially supporting this research under Contract No. NSC89-2213-E-009-015. The authors also express their gratitude to the reviewers for their valuable comments and suggestions.

References

1. P. J. Cheng & W. P. Yang (1999) A new content-based access method for video databases. *Information Science: An International Journal* **118,** 37–73.
2. M. L. Cascia & E. Ardizzone (1996) JACOB: just a content-based query system for video databases. In: *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, Atlanta, Georgia*, pp. 7–10.
3. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele & P. Yanker (1995) Query by image and video content: the QBIC system. *IEEE Computer* **28,** 23–32.
4. H. J. Zhang, J. Wu, D. Zhong & S. W. Smoliar (1997) An integrated system for content-based video retrieval and browsing. *Pattern Recognition* **30,** 643–658.
5. T. Arndt & S. K. Chang (1989) Image sequence compression by iconic indexing. In: *IEEE Workshop on Visual Languages, Rome, Italy*, pp. 177–182.
6. C. W. Chang & S. Y. Lee (1997) A video information system for sport motion analysis. *Journal of Visual Languages and Computing* **8,** 265–287.
7. S. F. Chang, W. Chen, H. J. Meng, H. Sundaram & D. Zhong (1998) A fully automated content-based video search engine supporting spatiotemporal queries. *IEEE Transactions on Circuits and Systems for Video Technology* **8,** 602–615.

8. P. J. Cheng & W. P. Yang (1999) Querying video contents by motion example. In: *Proceedings of International Symposium on Database Applications in Non-Traditional Environments*, *Kyoto*, *Japan*, pp. 291–297.
9. J. D. Courtney (1997) Automatic video indexing via object motion analysis. *Pattern Recognition* **30,** 607–625.
10. B. Gunsel, A. M. Tekalp & P. J. L. V. Beek (1998) Content-based access to video objects: temporal segmentation, visual summarization, and feature extraction. *Signal Processing* **66,** 261–280.
11. F. J. Hsu, S. Y. Lee & B. S. Lin (1998) Video data indexing by 2D C-trees. *Journal of Visual Languages and Computing* **9,** 375–397.
12. S. Y. Lee & H. M. Kao (1993) Video indexing—an approach based on moving object and track. In: *Storage and Retrieval for Image and Video Databases*, *SPIE* **1908,** pp. 25–36.
13. J. Z. Li, M. T. Ozsu & D. Szafron (1997) Modeling of moving objects in a video database. In: *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, *Ottawa*, *Ontario*, *Canada*, pp. 336–343.
14. E. Sahouria & A. Zakhor (1997) Motion indexing of video. *IEEE International Conference on Image Processing* **2,** pp. 526–529.
15. K. Shearer, D. Kieronska & S. Venkatesh (1997) Resequencing of video using spatial indexing. *Journal of Visual Languages and Computing* **8,** 193–214.
16. A. Yoshitaka, Y. I. Hosoda, M. Yoshimitsu, M. Hirakawa & T. Ichikawa (1996) VIOLONE: video retrieval by motion example. *Journal of Visual Languages and Computing* **7,** pp. 423–443.
17. N. Dimitrova & F. Golshani (1995) Motion recovery for video content classification. *ACM Transactions on Information Systems* **13,** 408–439.
18. P. J. Cheng & W. P. Yang (1999) An approach of modeling object motions for video libraries. In: *Proceedings of the Second Asian Digital Library Conference*, *Taipei*, *Taiwan*, pp. 226–242.
19. H. Jiang, A. S. Helal, A. K. Elmagarmid & A. Joshi (1998) Scene change detection techniques for video database systems. *Multimedia Systems* **6,** 186–195.
20. S. M. Smith (1998) ASSET-2: real-time motion segmentation and object tracking. *Real-Time Imaging* **4,** pp. 21–40.
21. M. Vazirgiannis, Y. Theodoridis & T. Sellis (1998) Spatio-temporal composition and indexing for large multimedia applications. *Multimedia Systems* **6,** 284–298.
22. N. Pissinou, I. Radev, K. Makki & W. J. Campbell (1998) A topological-directional model for the spatio-temporal composition of video objects. In: *Proceedings of Eighth International Workshop on Research Issues In Data Engineering*, *Orlando*, *Florida*, pp. 17–24.
23. T. D. C. Little & A. Ghafoor (1993) Interval-based conceptual models for time-dependent multimedia data. *IEEE Transactions on Knowledge and Data Engineering* **5,** 551–563.
24. Y. P. Wang & T. Pavlidis (1990) Optimal correspondence of string subsequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12,** 1080–1087.
25. V. N. Gudivada & V. V. Raghavan (1995) Design and evaluation of algorithms for image retrieval by spatial similarity. *ACM Transactions on Information Systems* **13,** 115–144.

Appendix A

Lemma 4.1. *Algorithm* 4.1 *produces a triangulate association mapping.*

Proof. Lemma 4.1 can be intuitively proved by showing that the following statements hold:

(a) $D(i,j) = D(i-1,j) + p(a_i, b_j)$ never generates a $TAF_{A\rightarrow B}$ such that $TAF(i-1) = [j-1,j]$ and $TAF(i) = [j,j]$, and

(b) $D(i,j) = D(i,j-1) + p(a_i, b_j)$ never generates a $TAF_{A\rightarrow B}$ such that $TAF(i-1) = [j-1,j-1]$ and $TAF(i) = [j-1,j]$.

Assume (a) is false. When $p\,(a_{i-1}, b_j) > 0$, $D(i,j) = D(i-1,j) + p(a_i, b_j) = (D(i-1,j-1) + p\,(a_{i-1}, b_j)) + p(a_i, b_j) \le D(i-1,j-1) + p(a_i, b_j)$, which is a contradiction. When $p\,(a_{i-1}, b_j) = 0$, $D(i-1,j) = D(i-1,j-1)$ and $TAF\,(i-1) = [\,j-1,j\,]$ can be reduced to $TAF(i-1) = [\,j-1,j-1\,]$. The same as (a), (b) also conflicts. Therefore, Algorithm 4.1 gives a TAF. □

Theorem 4.1. *Given two sequences* $A = a_1 a_2 a_3, \ldots, a_n$ and $B = b_1 b_2 b_3, \ldots, b_m$, *a triangulate association function* (TAF) *of* A *and* B *and a relation*

$$D(i,j) = \min \begin{cases} D(i-1,j-1) + p(a_i, b_j) \\ D(i-1,y) + p\,(a_i, b_{y+1}, \ldots, b_j) \ \text{for}\ y = 1, \ldots, j-2 \\ D(x,j-1) + p(b_j, a_{x+1}, \ldots, a_i) \ \text{for}\ x = 1, \ldots, i-2 \end{cases}$$

with $D(1,1) = p\,(a_1, b_1)$, $D(i,1) = \sum_{\forall i} p\,(a_i, b_1)$ *and* $D(1,j) = \sum_{\forall j} p\,(a_1, b_j)$, *then*

$$\forall\, TAF,\ D\,(n,m) \le \sum_{\forall i(|TAF_{A\to B}(a_i)| \ge 1)} p\,(a_i, TAF_{A\to B}(a_i))$$

$$+ \sum_{\forall j\,(|TAF_{B\to A}(b_j)| > 1)} p\,(b_i, TAF_{B\to A}\,(b_i))$$

Namely, $D\,(n, m)$ *is optimal.*

Proof. We prove it by double induction over i and j $(1 \le i \le n, 1 \le j \le m)$.

Basis. $D(1,1) = p\,(a_1, b_1)$. For all i,j, $D\,(i,1) = \sum_{\forall i} p\,(a_i, b_1)$ and $D\,(1,j) = \sum_{\forall j} p\,(a_1, b_j)$. The statement is true by definition of D.

Induction. Assume that $D(i,j)$ is optimal for $1 \le i \le n-1$ and $1 \le j \le m-1$. We wish to show by induction that $D(n, m)$ is optimal.

Case 1. $D\,(n, m) = D\,(n-1, m-1) + p\,(a_n, b_m)$. If $D\,(n, m)$ is not optimal, there exists a $D'\,(n, m)$ such that $D\,(n, m) = D\,(n-1, m-1) + p\,(a_n, b_m) > D'\,(n, m) = D'\,(n-1, m-1) + p(a_n, b_m)$. Thus, we have $D\,(n-1, m-1) > D'\,(n-1, m-1)$, which is a contradiction. Similarly, the other cases $(D(n,m) = D(n-1,y) + p\,(a_n, b_{y+1}, \ldots, b_m)$ *for* $y = 1, \ldots, m-2$ and $D\,(n-1, m) = D(x, m-1) + p\,(b_m, a_{x+1}, \ldots, a_n)$ *for* $x = 1, \ldots, n-2)$ satisfy the inequality. Therefore, by definition $D\,(n, m)$ is optimal. □

Lemma 4.2. *Given two sequences* $A = a_1 a_2 a_3, \ldots, a_n$ *and* $B = b_1 b_2 b_3, \ldots, b_m$, *if* $p\,(a_i, b_u b_{u+1}, \ldots, b_v) = \sum_{j=u,..,v} p\,(a_i, b_j)$ *and* $p\,(b_j, a_u a_{u+1}, \ldots, a_v) = \sum_{i=u,..,v} p\,(a_i, b_j)$ *then* $FMP^{OTCS/w}_{TAF}(A, B) = FMP^{OTCS/o}_{TAF}(A, B)$. *In other words,* OTCS/o *is a special case of* OTCS/w.

Proof. According to Lemma 4.1 and Algorithm 4.1, if $p(a_i, b_u b_{u+1}, \dots, b_v) = \sum_{j=u} p(a_i, b_j)$ and $p(b_j, a_u a_{u+1}, \dots, a_v) = \sum_{i=u..v} p(a_i, b_j)$ then

$D(i,j) = \min$

$$\begin{cases} D(i-1,j-1) + p(a_i, b_j) \\ D(i-1,j) + p(a_i, b_j) = \min(D(x,j-1) \\ \qquad\qquad\qquad + p(b_j, a_{x+1} \dots a_i) \text{ for } x = 1, \dots, i-2), \text{ where} \\ D(i,j-1) + p(a_i, b_j) = \min(D(i-1,y) \\ \qquad\qquad\qquad + p(a_i, b_{y+1} \dots b_j) \text{ for y} = 1, \dots, j-2) \end{cases}$$

$$\begin{aligned} &D(i-1,j) + p(a_i, b_j) \\ &= \min(D(i-2,j-1) + p(a_{i-1}, b_j), D(i-2,j) + p(a_{i-1}, b_j)) \\ &\quad + p(a_i, b_j) = \cdots \\ &= \min(D(x,j-1) + p(b_j, a_{x+1} \dots a_i) \textit{ for } x = 1, \dots, i-2). \end{aligned}$$

Notice that $D(i-1,j) \neq D(i-1,j-1) + p(a_{i-1}, b_j)$; otherwise, the mapping between A and B will not be a TAF based on Lemma 4.1. Similarly, $D(i, j-1) + p(a_i, b_j) = \min(D(i-1,y) + p(a_i, b_{y+1} \dots b_j) \quad \textit{for} \quad y = 1, \dots, j-2)$ also holds. Thus, OTCS/w is equivalent to OTCS/o when $p(a_i, b_u b_{u+1}, \dots, b_v) = \sum_{j=u,..,v} p(a_i, b_j)$ and $p(b_j, a_u a_{u+1}, \dots, a_v) = \sum_{i=u,..,v} p(a_i, b_j)$. □

Theorem 4.2. *Given two sequences* $A = a_1 a_2 a_3, \dots, a_n$ *and* $B = b_1 b_2 b_3, \dots, b_m$, *a triangulate association function* (TAF) *of* A *and* B, *and a relation*

$$D(i,j) = \min \begin{cases} D(i-1,j-1) + p(a_i, b_j) \\ D(i-1,j) + p(a_i, b_j) \\ D(i,j-1) + p(a_i, b_j) \end{cases}$$

with $D(1,1) = p(a_1, b_1)$, $D(i,1) = \sum_{\forall i} p(a_i, b_1)$ *and* $D(1,j) = \sum_{\forall j} p(a_1, b_j)$, *then* $\forall$ *TAF*, $D(n,m) \leq \sum_{\forall i(j \in TAF(i))} p(a_i, b_j)$. *That is*, $D(n,m)$ *is optimal.*

Proof. This theorem results from Theorem 4.1 and Lemma 4.2. □

Appendix B

Notation	Description
$[x,y]$	interval of numbers from x to y
disjoin	disjoint combination of an interval set

v	video stream
s	scene
τ	the set of objects in v
m	motion composition ($m = Q * R$)
q	motion sequence (Q = a set of q)
f	feature sequence
r	relation sequence (R = a set of r)
c^m	motion clip (ℓ = an ordered set of c^m)
c^f	feature clip (= an ordered set of c^f)
c^r	relation clip (ϑ = an ordered set of c^r)
$[x.\alpha, x.\beta]$	frame interval of x, where $x = s, q, f, r, c^m, c^f$ or c^r
$x.\delta$	frame-interval size of x, where $x = s, q, f, r, c^m, c^f$ or c^r
σ^x	normalization threshold, where x = *traj*, *hist* or *size*
$e(\ )$	evaluation function
$\nabla(\ )$	dissimilarity function
I_i	the ith image frame (v = an ordered set of I)
σ	selection operation of m
π	projection operation of m
$\otimes$	join operation of m's
$\cup, \cap, -$	union/intersection/difference operation of m's
$p(\ )$	penalty function
$TAF(\ )$	triangulate associate function
$OTCS/o$	optimal triangulate correspondence of sequences without merging
$OTCS/w$	optimal triangulate correspondence of sequences with merging
$D_y^x(A, B)$	distance between sequences A and B, where X = OTCS/o or OTCS/w and Y = full or partial
$T[u_i, u_j]$	an arbitrary fragment of a trajectory T from point u_i to point u_j
$AP(T_1, T_2)$	average distance between T_1 and T_2 based on vertices' projections
$MP(T_1, T_2)$	maximum distance between T_1 and T_2 based on vertices' projections
$AE(T_1, T_2)$	average distance between T_1 and T_2 based on equidistant points
$ME(T_1, T_2)$	maximum distance between T_1 and T_2 based on equidistant points
R_k	recall of the top k objects
P_k	precision of the top k objects