Journal of Visual Languages and Computing (1997) 8, 265–287

A Video Information System for Sport Motion Analysis

CHUEH-WEI CHANG AND SUH-YIN LEE

Institute of Computer Science and Information Engineering, National Chiao Tung University, 1001 Ta Hsueh Road, Hsinchu, Taiwan 30050, R.O.C., cwchang@info1.csie.nctu.edu.tw

Received 12 July 1996; revised 20 January 1997; accepted 29 January 1997

In this paper, we present several important methods in a video information system for sport motion analysis. We first describe an efficient method for computing the position of one or more moving objects in a video sequence. We utilize computer vision routines for both target object tracking and feature value extraction. Then, we propose a multiple subsequence matching mechanism for comparing the difference in object motion between two video sequences. A mapping function, penalty table and the overall distance between two video sequences have been defined. We use the prominent feature points as our segmentation positions and separate the sequence into several subsequence groups. After an alignment process, we can find the mapping of the best matching and get an evaluation report between the two video sequences for motion analysis. The video information system described in this paper has successfully analysed several examples of sport motion, and is representative of techniques commonly used in practice.

1. Introduction

WITH RAPID advances in data storage, image processing, telecommunication and data compression, video has become an important component in the new generation of information systems. Video data contain a large amount of spatial and temporal information. They can provide more information than text, graphics and even image can do. The information related to the position, distance, temporal and spatial relationships are included in the video data implicitly. Especially, the changes of video objects in equally divided temporal intervals are quite useful for motion analysis and cannot be provided by other media easily.

The use of image sequences in areas such as entertainment, visual communication, multimedia, education [1], medicine, motion prediction [2] and scientific research is consistently increasing as the use of television and is becoming more and more common. For example, interactive video [3] makes possible a detailed study of real-life events such as the motion of a volleyball. The applicability of digital video for both consumer and professional products, along with the availability of fast processors and memory at reasonable costs, has been a major driving force behind this growth. However, many of the methods developed to date for motion analysis are still computationally expensive.

Therefore, a fast, easy-to-use and cost-effective video information system becomes an important requirement for the application of motion analysis. Recently, there is an

1045-926X/97/030265+23 $25.00/0/vl970043

increasing interest in video databases that store and retrieve video clips [4, 5]. Most of these available video databases offer retrieval functions where users specify key words such as titles, attributes or actors. But, they can only provide to the users suitable key words to be selected according to the purpose of retrieval. If this video database is used for scientific computation, no one knows what its internal value will be and how a query can be submitted. Therefore, providing key examples [6] becomes an important issue for the video information systems. Furthermore, designing a fast video matching and indexing mechanism is another important issue in video information systems.

In this paper, we present the methodology of designing a video information system for sport motion analysis. In Section 2, we give a brief introduction to the video information system and sport motion analysis. In Section 3, we describe an efficient method for extracting the position of one or more moving objects in a video sequence. In Section 4, we propose several matching models for comparing the differences in object motion between two video sequences. In Section 5, we provide a sophisticated video content segmentation and indexing mechanism. In Section 6, we present our video information prototype system and sport motion analysis subsystem along with several query examples. The concluding remarks are given in Section 7.

2. Video Information System and Sport Motion Analysis

2.1. Video Information System

A general *video information system* [7] is an information system that manages the video input, video processing, video query, video storage and video indexing to provide a collection of video data for easy access by a large number of users. Also, a *video database* [8] is the core of a video information system, which manages video, including retrieving, inserting and editing video data, maintaining video indices and processing queries.

We have designed a video information system for multiple purpose use [9–11] before we enhance and apply this system to the domain of sport motion analysis. This video information system has the capability to analyse and retrieve video sequences based on the video contents [12]. This content-based system allows users to specify queries using actual video data in addition to the traditional textual annotation. This process is typically called query-by-example (QBE).

In order to focus on the problem of motion analysis in volleyball, we discuss only some of the major mechanisms in our video information system. The key component of our motion analysis process is based on object extraction and temporal feature matching procedures. The object extraction approach calculates the differences between successive frames and then tracks the area of large difference to determine target objects. An object in a video frame is searched for in a succeeding video frame. If the same object is found, the feature values are calculated from the object. The temporal feature matching mechanism provides a multiple subsequence matching algorithm for comparing the difference in object motions between two video sequences. We use the prominent feature points as segmentation positions and separate the sequence into several subsequence groups. After an alignment process, we can find the mapping of the best match, and get an evaluation report for the two video sequences for motion analysis.

2.2. Sport Motion Analysis by Computer Vision

Sport research, in general, has increased dramatically over the past few years [13]. Recent research has made a tremendous contribution in improving a player's conditioning program, developing material strategies, designing better equipment and understanding performance techniques. The Germans are renowned for their research and the Americans are well-known for their application of research to track and field.

We believe that imitation is one of the best ways to get improvement in the learning stage. Correct body posture is extremely important for performance in sports. By using the correct posture, a player can perform skills more efficiently and with less chance of injury. For example, in a volleyball court, due to the many variables associated with spiking, the timing of spiking is difficult to determine and the success of spiking requires hours of practice. By comparing the posture frame-by-frame, a player can realize where he goes wrong and what the best choice is. That is, the process of visual comparison via image sequences is one of the best ways to improve the skills of volleyball players. Furthermore, good prediction is now important winning a game in any competition. If we collect enough image sequence samples, we can not only adjust the posture in a practice, but can also predict the possible actions in a series of movements. If we can utilize the spatio-temporal characteristics of image sequences, we can have the capability to analyse the activities of movement in the volleyball court. For example, we can extract the jumping altitude and the coordinates of strike hand from an image sequence.

In most of the motion analysis experiments, researchers use a tripod-mounted high-speed film camera that can take film shots at 100 frames per second to record the motion of the object as it moves across the field of view, and/or use a large number of sensors and wires to capture the slight movement of target objects. One of the most popular techniques for collecting object positions is to place a sheet of clear acetate transparency on the video screen, mark points of interest in the picture, redraw on a graph paper and then read off coordinates for the measurement scale for each of the points. Marking points of interest on each frame is usually very time-consuming during a traditional motion analysis process, because it involves measuring extremely small dots on a photograph. Furthermore, this experimental equipment is very expensive and can also restrict the performance of target objects. Therefore, we attempt to set up a new sport motion analysis environment with the help of digital video, a personal computer and a video information system.

The essence of this scientific sport research is to obtain spatio-temporal information for the player. We can easily realize that information about the position and velocity of objects is important to the sport motion analysis process. This information can contribute significantly to posture interpretation: object trajectories may be analysed, important events may be noted and future situations may be predicted.

Very little research has been done in sport motion analysis by computer-based video. The VideoGraph [14] provided a good means for physics instruction, but it did not provide multiple view ports in its user interface or matching model for temporal data analysis. Therefore, designing a video information system that can analyse the motion phenomena, figuring out the variation and collecting a large amount of data for searching and prediction, are important steps toward success.

3. Motion Object Feature Extraction

3.1. Object Tracking and Feature Extraction

Object tracking searches the video sequence for the appearances of a specified object. It requires identifying the specified object in each frame of the video sequence. Several methods are known for this identification [15, 16]. For example, template matching is one of the famous methods. However, we have to adopt a method satisfying the condition of robustness against the changes of the object size, color, location and direction.

In the motion object feature extraction mechanism, we use both movement and color information of the object. Both types of information can provide good image cues for object segmentation. Therefore, the approach of motion object extraction and analysis in video sequences can be divided into five steps as shown in Figure 1: (1) specifying the target object by establishing a color table and object segmentation, (2) providing bilevel video frame difference, (3) removing small frame differences caused by small motions and random noise, (4) finding the major areas with significant movement and finding the position of target object via color table and matching process and (5) extracting feature values of target objects and generating the motion vector as a reference for the next video frame.

The purpose of the object color table is to provide information regarding the colors on target objects, to find out the positions of the target objects and to segment these target objects from the video frames. This color table can be constructed manually by a user by selecting a specific area containing the target object or by picking up a group of colors. In a real-world video frame, there is little chance that the color of a pixel is exactly the same as its neighbor. Due to the influence of shades, highlights and other complicated illumination conditions, pixels belonging to the same region may show complex color distribution whereby erroneous segmentation may be obtained. Being aware of this limitation, an algorithm to detect regions, based on object color clustering, is designed and applied to supplement the object-tracking procedure [10].

The approach for providing bilevel video frame difference between two video frames k and $k+1$ is to compare the two images pixel by pixel. Suppose that we have a reference image containing only stationary components. Comparing this reference image against a subsequent image having the same background but including a moving object results in the difference image which contains only nonzero entries corresponding to the nonstationary components with stationary components being canceled.

A difference image, $d_{k,k+1}$, between video frames k and $k+1$ may be defined as the pixel to pixel difference in the two successive video frames:

$$d_{k,k+1}(i,j) = \begin{cases} 1 & \text{if } \|f(i,j,k) - f(i,j,k+1)\| > \eta, \quad 0 \le i < x,\ 0 \le j < y \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where η is a difference threshold, x and y are the frame resolutions in the X- and Y-axis of the image plane coordinate system, respectively, and $f(i,j,k)$ is the color intensity in coordinate (i,j) of the video frame k. The pixel-to-pixel difference, $\|f(i,j,k) - f(i,j,k+1)\|$, is a distance measure, such as the Euclidean distance in RGB color space. Note that $d_{k,k+1}(i,j)$ equals 1 at coordinates (i,j) only if the intensity-level difference between the two images is appreciably different at those coordinates, as determined by the difference threshold η.

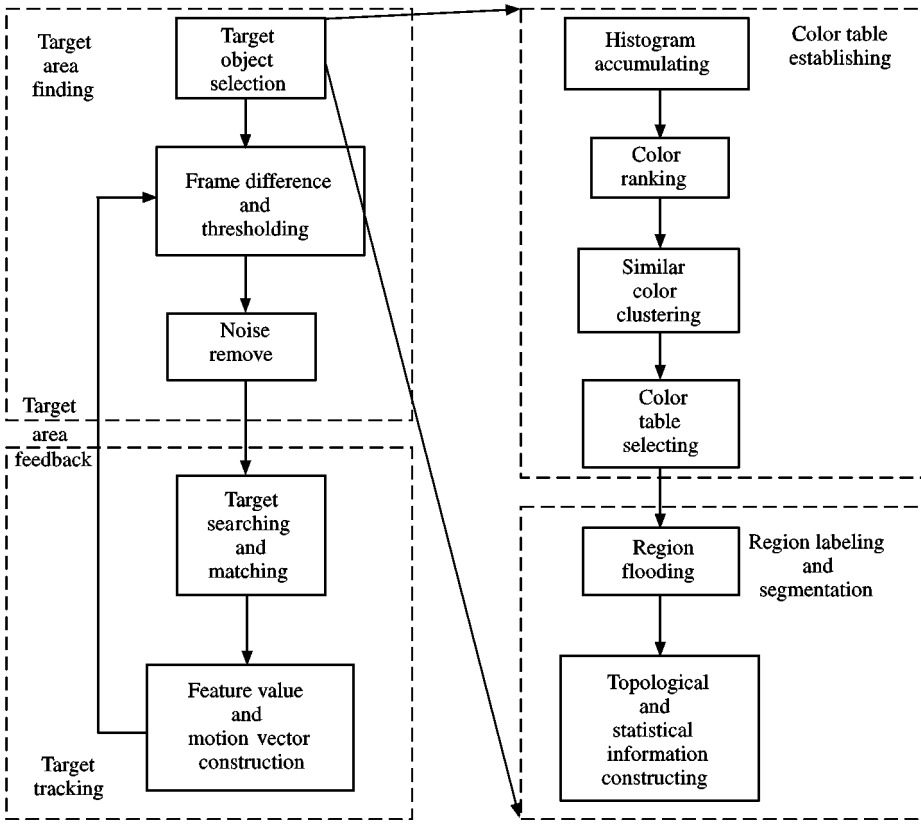
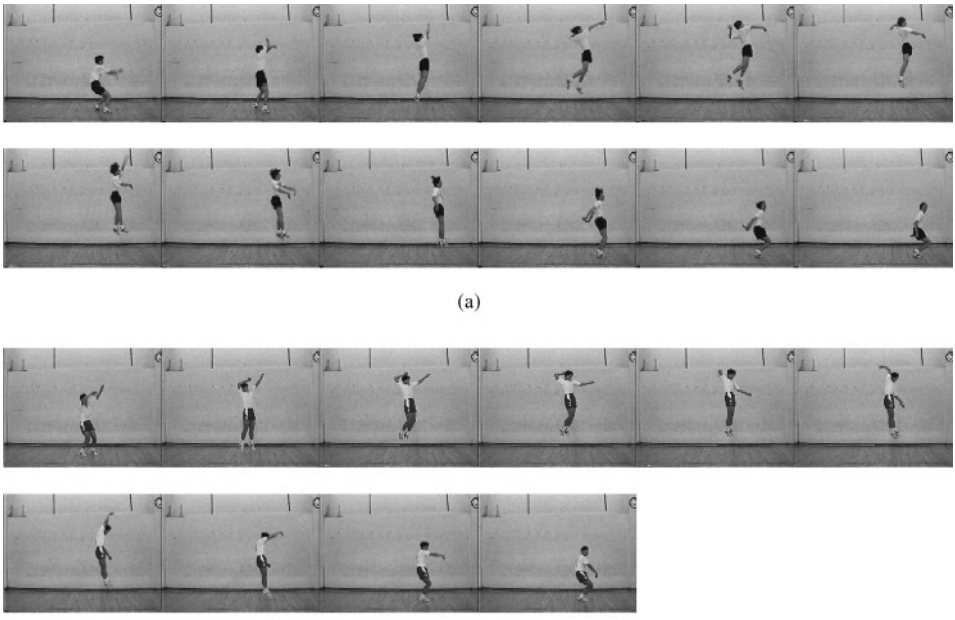


Figure 1. Object tracking procedures

In motion analysis, all pixels in $d_{k,k+1}(i, j)$ with value 1 are considered to be of significant object motion. This approach is applicable only if the two images are registered and the illumination is relatively constant within the bound of η. Furthermore, 1-valued entries in $d_{k,k+1}(i, j)$ sometimes arise due to noise in practice, and these entries are typically isolated in the difference image. A simple approach to remove these isolated points is to form 4- or 8-connected regions [17] in $d_{k,k+1}(i, j)$ and then ignore any region that has less than a predetermined number of connected points. Although it may result in ignoring small moving objects, this approach has the better possibility that the remaining entries in the difference image are actually the results of motion. The way we remove the small movements is to run a mask over the difference image such as the traditional noise removal algorithm [17]. The mask size depends on the area of the region we wish to remove from the image.

Then, we can enclose those remaining large movement regions by a minimum bounding rectangle (MBR). An MBR is the smallest rectangle enclosing a large movement region. We use the MBR as the selected region of our target area. The colors in a real-world video frame are complicated and hence it is hard to extract the edge information from the video sequence. Thus, we use a more efficient way to extract



(a)

(b)

Figure 2. Two video sequences of volleyball attack approach: (a) target sequence (frame 18–29 only) and (b) sample sequence (frame 13–22 only)

a specific object of the video frame. That is, we search the selected region in the video frame and the specific object will appear in a major area filled with specified colors according to the object color table. The process of target area selection will greatly reduce the searching range, the possibility of mismatch and computation time. A second MBR (SMBR) is generated by enclosing the specific object. The coordinate of the SMBR will be a good reference for the object position and the motion vector.

3.2. An Example for Trajectory Extraction

We use a volleyball spike (attack) approach shown in Figure 2 as our example. We take these video sequences simply using a Hi-8 home video camera in a gym and digitize the video sequences into video data files by a video capture card installed in a personal computer. We also specify an origin of coordinates, calibrate the image by selecting an object of known size, such as the baseline mark from a frame and then enter its size and units of measurement.

The basic approach to spike is described as follows: for a high outside set, the attacker begins on the attack line, waits for the set and then moves toward the set. Approach the net, covering the distance with as few steps as possible. Make a two-footed takeoff by planting the right heel first and closing with left foot or by taking a hop onto both feet. As the attacker plants both heels first to change forward momentum into upward momentum, he swings arms to prepare for a jump. Swing both arms forward

Table 1. Target sequence feature values

Frame no.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Jumping	212	212	212	212	212	212	212	212	212	212	212	212	212	212	212
Altitude (cm)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Centroid *x*	33	33	34	36	41	45	49	57	61	71	85	98	116	128	143
Centroid *y*	144	146	146	146	147	144	142	145	144	147	150	153	152	148	152
Hand *x*	28	25	25	28	38	49	68	81	93	110	118	92	83	99	121
Hand *y*	156	156	157	156	159	156	149	131	120	125	155	169	130	106	100
Hand *y* (cm)	83	83	82	83	79	83	94	121	137	130	85	64	122	158	167

Frame no.	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Jumping	212	212	212	212	209	183	169	159	159	170	180	202	212	212	212
Altitude (cm)	0	0	0	0	4·47	43·23	64·11	79·02	79·02	62·62	47·71	14·91	0	0	0
Centroid *x*	158	178	188	202	210	218	223	223	224	230	239	243	244	245	251
Centroid *y*	158	160	153	126	104	89	80	79	86	96	117	145	157	155	146
Hand *x*	130	190	230	222	206	193	186	203	264	259	231	217	224	252	269
Hand *y*	136	181	107	75	61	52	44	34	39	103	133	135	161	169	149
Hand *y* (cm)	113	146	157	204	225	239	250	265	258	163	118	115	76	64	94

Table 2. Sample sequence feature values

Frame no.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Jumping	212	212	212	212	212	212	212	212	212	212	212	212	212	195	170
Altitude (cm)	0	0	0	0	0	0	0	0	0	0	0	0	0	25·37	62·62
Centroid *x*	40	41	45	52	59	73	81	97	112	128	143	155	164	174	176
Centroid *y*	142	141	143	145	149	151	148	145	145	154	160	156	143	178	101
Hand *x* 1–15	33	45	49	63	68	57	36	52	79	85	115	179	89	173	169
Hand *y* 1–15	156	155	149	155	160	161	138	100	93	117	171	153	179	61	50
Hand *y* (cm)	83	85	94	85	78	76	110	167	177	142	61	88	49	225	242

Frame No.	16	17	18	19	20	21	22	23	24
Jump 16–30	164	167	172	181	201	212	212	212	212
Altitude	71·56	67·09	59·64	46·22	16·40	0	0	0	0
Centroid *x*	182	185	188	192	199	203	206	207	208
Centroid *y*	89	87	95	104	123	146	155	154	148
Hand *x*	170	169	172	214	238	246	234	225	226
Hand *y*	48	48	32	37	96	126	163	171	145
Hand *y* (cm)	245	245	268	261	173	128	73	61	100

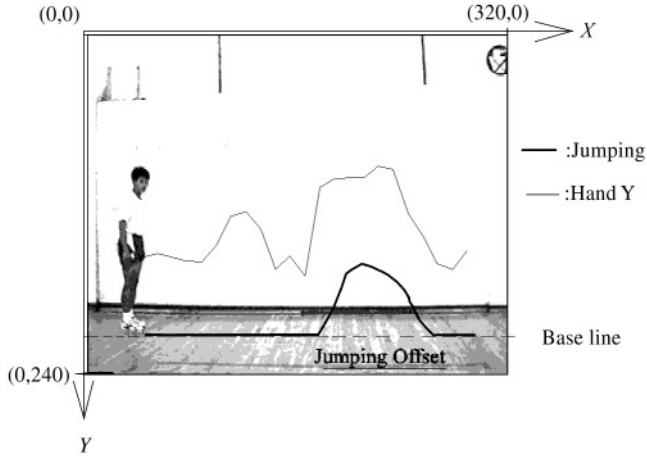


Figure 3. Moving path in the image coordinate system

and reach high toward the set as the attacker jumps straight up into the air. Draw hitting arm back, elbow high and hand close to ear. As the attacker swings at the ball, he drops the nonhitting hand quickly to his waist.

By using the object-tracking procedure stated in Section 3.1, we can get a series of feature values about the jumping altitude, position of preference hand and position of waist as shown in Tables 1 and 2.

After getting the origin of coordinates and the distance between two pixels via the image calibration procedure, we transform the coordinates of a series of object feature points into the world coordinates. In this case, we ignore the small errors due to perspective projection and camera lens distortion. If the baseline is parallel to the X-axis of the image plane, we transform the pixel-based image coordinates into centimeter-based world coordinates by a simple formula as follows:

$$C = (T - 1)R \tag{2}$$

where R is the unit distance in centimeter (distance between two pixels), T the number of pixels in image coordinate system and C the distance in centimeter in the world coordinate system. The changes of hand and heel positions are shown in Figure 3 and can be drawn as a function of frame and pixel distance as shown in Figure 4.

4. Content Alignment and Matching

4.1. Measurement of Distance and Matching Model

After the feature extraction process, we can compare the target video with a standard sample video and evaluate the difference between them. First, we need to define the measurement of the frame distance and the sequence distance between two video sequences for a specific feature.

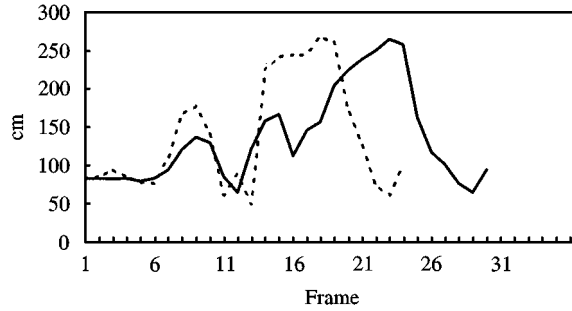


Figure 4. The hand swing position in the Y-axis. Dotted line is sample, solid line is target

In this volleyball spike example, we use the absolute difference in the feature values of two video frames as the frame distance. If two video sequences are similar in feature values and are of same duration, the sequence distance will be the summation of individual frame distance between corresponding frames. For the target sequence $A = a_1 a_2 \ldots a_n$, the sample sequence $B = b_1 b_2 \ldots b_m$, where n and m are the lengths of the sequences A and B, respectively. If $m = n$, the sequence distance can be simply obtained by

$$\sum_{i=1}^{m} |a_i - b_i| \tag{3}$$

We can also get a *mapping function* $F : [m] \to [n]$. If $m = n$, it will be a one-to-one and onto mapping function $F(i) = i$, as shown in Figure 5.

In practice, it is hard to shoot video precisely on the starting position of a series of movements even with a postprocess of editing. Furthermore, even if we can shoot video at a fixed rate, e.g. 30 frames per second, we still do not know where the starting frame of each target movement is in a video sequence. A point can be matched to any point in a sequence having the same value. This is the one-dimensional aperture problem in motion analysis community. Therefore, the position correspondence between two video sequences is a crucial task yet to be resolved.

The comparison between two video sequences is not just simply a fixed length template matching as the shift matching model shown in Figure 6. For each kind of

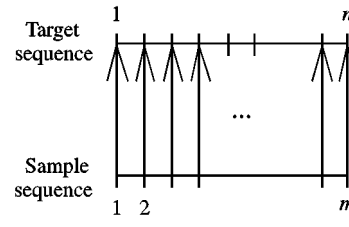


Figure 5. Exact matching model

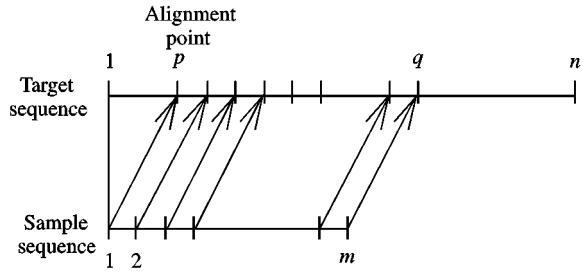


Figure 6. Shift matching model

video sequence matching model, two things need to be solved before calculating the sequence distance: the video frame alignment point and the video frame correspondence. That is, if $m < n$ and $q - p + 1 \neq m$ occur, where should we start the distance measurement? Where will be the left and right alignment points, p and q, respectively? How can we get the mapping function? What is the mapping function $F : [(1, 2, \ldots, m)] \rightarrow [(p, p + 1, \ldots, q])$?

4.2. Equal Length Content Alignment

A straightforward way to find a shift alignment point between two video sequences is to use a pattern-matching algorithm [18]. Therefore, we design an approximate pattern-matching algorithm, as shown in Algorithm 1, to provide the mechanism for content alignment. This approximate matching algorithm is an extended version of the KMP algorithm [19].

In our approximate matching algorithm, we change the finite symbolic representation of KMP algorithm into an infinite representation of real numbers. Furthermore, we replace the similarity relation ($\sim$) with the equality sign ($=$) defined in the original KMP algorithm. If $|a - b| < \tau$ is denoted as $a \sim b$, we say that a and b are similar and treated as a matched value, where a is in sequence A and b is in sequence B. If the

---

Algorithm 1. *Approximation pattern matching*
Input. A target sequence *target*$[1 \ldots n]$, a sample sequence *sample*$[1 \ldots m]$ and a slide table *next*$[1 \ldots m]$.
Output. Locations of matched pattern in target sequence.
Method.

```
j = k = 1;
while j <= m and k <= n do
begin
   while j > 0 and target[k] not ~ sample[ j ] do j = next[ j ];
   k = k + 1; j = j + 1;
   if j > m then j = 1; print matched pattern at position k − m + 1;
end;
```

End-of-Algorithm Single pattern approximation matching.

Algorithm 2. *Slide table construction*
Input. A matched pattern *sample*[1 . . *m*].
Output. A slide table *next*[1 . . *m*].
Method.

```
j = 1; r = 0; next[1] = 0;
while j < m do
begin
   while r > 0 and sample[ j ] not ~ sample[ r ] do t = next[ r ];
   r = r + 1; j = j + 1;
   if sample[ j ] ~ sample[ r ]
   then next[ j ] = next[ r ];
   else next[ j ] = r;
end;
```

End-of-Algorithm Slide table construction.

difference of two corresponding feature values in these two sequences is greater than the similarity threshold τ, we need to shift the sample sequence to the next check position with the help of slide table as the original KMP algorithm. If τ is set to be zero, this approximate matching becomes an exact matching algorithm.

The input target sequence is in an array *target*[1 . . *n*], and the sample sequence appears in *sample*[1 . . *m*]. Let k and j be integer variables such that *target*[k] denotes the current target sequence value and *sample*[j] denotes the corresponding sample sequence value. The slide table *next*[ ] can be computed as in Algorithm 2.

Example 1. Given two sequences, target sequence $A = \{36{\cdot}1, 87{\cdot}3, 75{\cdot}9, 34{\cdot}2, 87{\cdot}6, 36{\cdot}6, 24{\cdot}5, 59{\cdot}2, 35{\cdot}2, 86{\cdot}9, 35{\cdot}3, 45{\cdot}5, 35{\cdot}2, 24{\cdot}5, 71{\cdot}4\}$ and sample sequence $B = \{34{\cdot}2, 89{\cdot}2, 35{\cdot}1, 47{\cdot}1, 35{\cdot}4, 23{\cdot}7\}$ as shown in Figure 7, with the absolute difference as

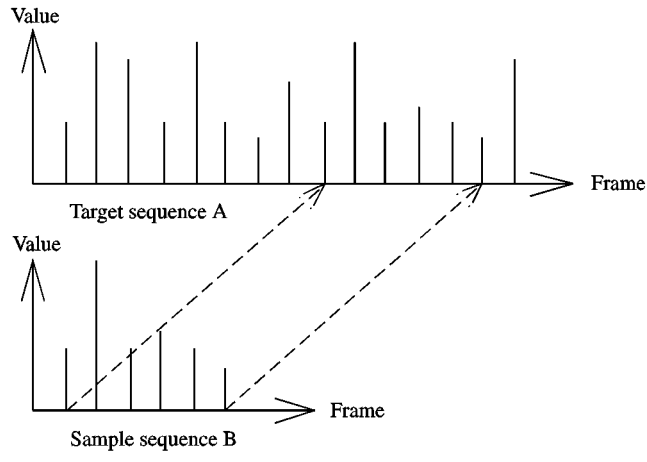


Figure 7. An example of equal length content alignment

Target sequence A

36.1, 87.3, 75.9, 34.2, 87.6, 36.6, 24.5, 59.2, 35.2, 86.9, 35.3, 45.5, 35.2, 24.5, 71.4

Sample sequence B 34.2, 89.2, 35.1, 47.1, 35.4, 23.7

↑

Figure 8. A match of target and sample sequence

k =	1	2	3	4	5	6
sample[k] =	34.2	89.2	35.1	47.1	35.4	23.7
next[k] =	0	1	0	2	0	2

Figure 9. The slide table of a sample sequence

a similarity measure. The length of A is 15 and the length of B is 6. According to the pattern-matching method in Algorithm 1, a matching between target sequence A and sample sequence B occurs at the position of the ↑ mark shown in Figure 8, where τ is 4.1. The slide table is shown in Figure 9.

After this process, we can get the mapping function $F : [(1, 2, \ldots, m)] \rightarrow [(p, p+1, \ldots, p+m-1)]$, if p is the starting point matched. Therefore, the sequence distance in this shift matching model can be defined by

$$\sum_{i=1}^{m} |a_{p+i-1} - b_i| \tag{4}$$

where p is the shift alignment point, and a_i and b_i are the feature values in the target and sample sequences, respectively. Although this extended KMP algorithm can find several candidates, we need only the closest one.

4.3. Complicated Content Correspondence

A more realistic and complicated situation is the one in which we need to skip some of the video frames, as shown in Figure 10 and then find the best-matched position and the best mapping. This situation is caused by the bad timing of movement (too fast or too slow). Unfortunately, the approximate pattern matching algorithm does not work when the case $q - p + 1 > m$ occurs.

Therefore, we use the algorithm in the optimal correspondence of string subsequences (OCS) [20] to solve our sequence-correspondence problem. We slightly change the definitions in the OCS algorithm to meet our requirements. The sequence distance between two sequences A' and B' is now redefined as:

$$S_{F'}(A', B') = (m' + n' - 2r)\tau + \sum_{\forall i, F'(i) \neq \perp} |a'_i - b'_{F'(i)}| \tag{5}$$

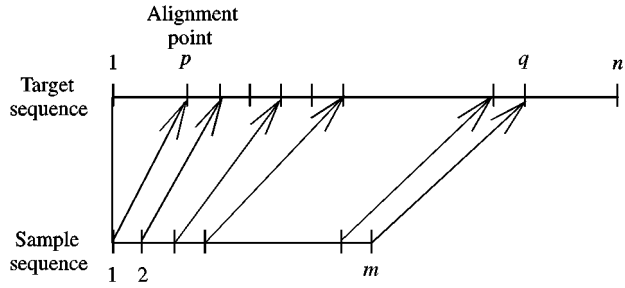


Figure 10. Complicated matching model

where m' and n' are the length of sequences A' and B', respectively, r is the number of matched pairs and τ the error tolerance. The best-matched distance between two sequences is defined as

$$D(A', B') = \min_{\forall F'} S_{F'}(A', B') \tag{6}$$

The new definition of mapping function: it is a mapping such that $F(i) = \perp$ or $(F(i) \neq \perp$ and $F(j) \neq \perp$ and $i < j) \rightarrow (F(i) < F(j))$. $\perp$ stands for the mismatch in a position. The minimum distance mapping can be expressed as follows. Given two sequences A' and B', find a mapping function F of A' and B' such that the sequence penalty is minimized. In order to find out the required mapping function F of A' and B', we build a minimum penalty table which accumulates the information of correspondence, and from the table we can trace back the correspondence and construct the desired F. The penalty table construction algorithm is depicted in Algorithm 3. The mapping function construction algorithm is shown in Algorithm 4. An example of a penalty table for sample sequence $\{268, 261, 173, 128, 73\}$ from frame 18 to frame 22 and target

---

Algorithm 3. *Penalty table construction*
Input. A target sequence *target*$[1..n]$, a sample sequence *sample*$[1..m]$, and the similarity threshold τ.
Output. A penalty table *penalty*$[1..n, 1..m]$.
Method.
penalty$[0, 0] = 0$;
for $i = 1$ to n do *penalty*$[i, 0] = i*\tau$;
for $j = 1$ to m do *penalty*$[0, m] = j*\tau$;
for $i = 1$ to n do
    for $j = 1$ to m do
  penalty$[i, j] = \min\{$*penalty*$[i-1, j-1] + \text{abs}($*target*$[i] -$ *sample*$[j]),$
                penalty$[i-1, j] + \tau$, *penalty*$[i, j-1] + \tau\}$;
return penalty$[n, m]$;
End-of-Algorithm Penalty table construction.

Algorithm 4. *Mapping function construction*
Input. A penalty table *penalty*[1 . . *n*, 1 . . *m*], and the similarity threshold τ.
Output. A mapping function set mapping[ ].
Method.

```
i = n; j = m;
while (i < > 0 and j < > 0) do
   begin
      if penalty[i, j] = penalty[i − 1, j] + τ then i = i − 1;
      else
         if penalty[i, j] = penalty[i, j − 1] + τ then j = j − 1;
         else
            begin
               add (i, j) to mapping[ ];
               i = i − 1; j = j − 1;
            end;
   end;
return mapping[ ];
```

End-of-Algorithm Mapping function construction.

sequence $\{265, 258, 163, 118, 115, 76\}$ from frame 23 to frame 28 is shown in Table 3, where τ is 40. The mapping function is $F: \{(18, 23), (19, 24), (20, 25), (21, 26), (\perp, 27), (22, 28)\}$. Frame 27 in target video is skipped in this mapping and the sequence distance is 69.

4.4. Multiple Subsequences Matching

When the frame numbers of two video sequences become large, the calculation of the penalty table will be time consuming. In order to reduce the computation time, we attempt to segment video sequences into multiple subsequence groups by multiple alignment points. Therefore, the matching model for multiple subsequences as shown in Figure 11 is more sophisticated than the complicated matching model. After presenting

Table 3. Penalty table of group S-7 and Q-6

(Frame) Value	Target	(23) 265	(24) 258	(25) 163	(26) 118	(27) 115	(28) 76
Sample	0	40	80	120	160	200	240
(18) 268	40	*3*	43	83	123	163	203
(19) 261	80	43	*6*	46	86	126	166
(20) 173	120	83	46	*16*	56	166	206
(21) 128	160	123	86	56	*26*	*66*	86
(22) 73	200	163	126	96	66	68	*69*

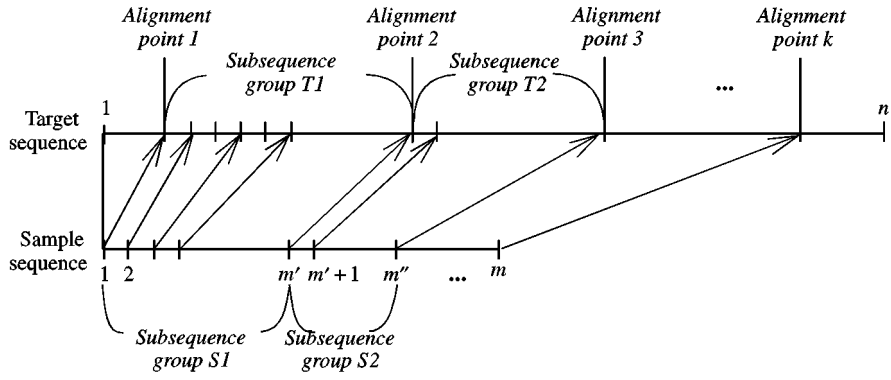


Figure 11. Matching model for multiple subsequences

the definitions of sequence distances and solving the case of the complicated matching model, we still need an efficient mechanism to overcome this most common and useful case—multiple subsequence alignment and matching problem.

If we can specify the corresponding subsequence groups, the overall distance can be defined as follows:

$$M(A, B) = \sum_{i=1}^{g} D_i(A^i, B^i) \tag{7}$$

where g is the number of groups and A^i and B^i are the ith subsequences of target sequence A and sample sequence B, respectively.

5. Content Segmentation and Indexing

5.1. Video Content Segmentation

The best way to find the alignment points is to segment the curve according to the prominent feature points in the sequences. In Chang and Lee [11], we proposed a segmentation mechanism by using curve features in video sequences. Several kinds of curve features can be found according to the changes of a sequence. These points with changing features are good positions for the alignment points. Basically, we classify curve features into four categories. They are: strongly up edge in a sequence, strongly down edge in a sequence, increase out of range and decrease out of range. By using this mechanism and properly selecting the threshold values, the sample and target curves of our volleyball hand swing position in the Y-axis can be segmented into subsequences, as shown in Tables 4 and 5. We use the segmented prominent points as the alignment points. After the segmentation process, a sequence can be divided into several subsequences enclosed by bounding boxes.

We use the most similar bounding boxes between target and sample sequences as the major subsequence alignment point and propagate the mapping of alignment points box-by-box into the left- and right-hand side of the sequence. For example, the Q-6 and

S-7 in Figure 12 are the major subsequence alignment points of target and sample sequences, respectively.

5.2. Time-series Video Indexing

The content-based indexing of a time-series sequence is a nontrivial and challenging problem [12, 21]. To speed up the specific feature point searching and subsequence alignment processing, we use B-tree [22] as our index structure [11]. We can search and find a possible major subsequence alignment point from the bounding box segmentation points with the help of this video indexing. We can use the prominent point as the key of index structure. The best choice of threshold value is dependent on the distribution of feature values and the density of prominent points.

Also, if we assume that two successive video frames in the same sequence are quite similar, the major portions of the video sequence remain unchanged, and the time intervals between frames are short enough that many changes do not occur at once. In particular, this means that the feature value of an object in the video sequence changes gradually. Several special characteristics exist: (1) two frames with a large difference tends to indicate that a special motion has occurred. In other words, an extremely large distance change means a special event has happened and (2) a special event can also be defined as the feature value in a single video frame in a specific predefined range.

Table 4. Prominent point information table for sample sequence

Segmentation point value	Subsequence group ID	Frame/time number	Box minimum	Box maximum	Frame/time offset
83	S-1	1	83	85	2
94	S-2	3	76	167	6
177	S-3	9	142	177	2
61	S-4	11	61	61	1
88	S-5	12	88	88	1
49	S-6	13	49	245	5
268	S-7	18	73	268	5
61	S-8	23	61	100	2

Table 5. Prominent point information table for target sequence

Segmentation point value	Subsequence group ID	Frame/time number	Box minimum	Box maximum	Frame/time offset
83	Q-1	1	82	121	8
137	Q-2	9	85	137	3
64	Q-3	12	64	158	3
167	Q-4	15	167	167	1
113	Q-5	16	113	250	7
265	Q-6	23	76	265	6
64	Q-7	29	64	94	2

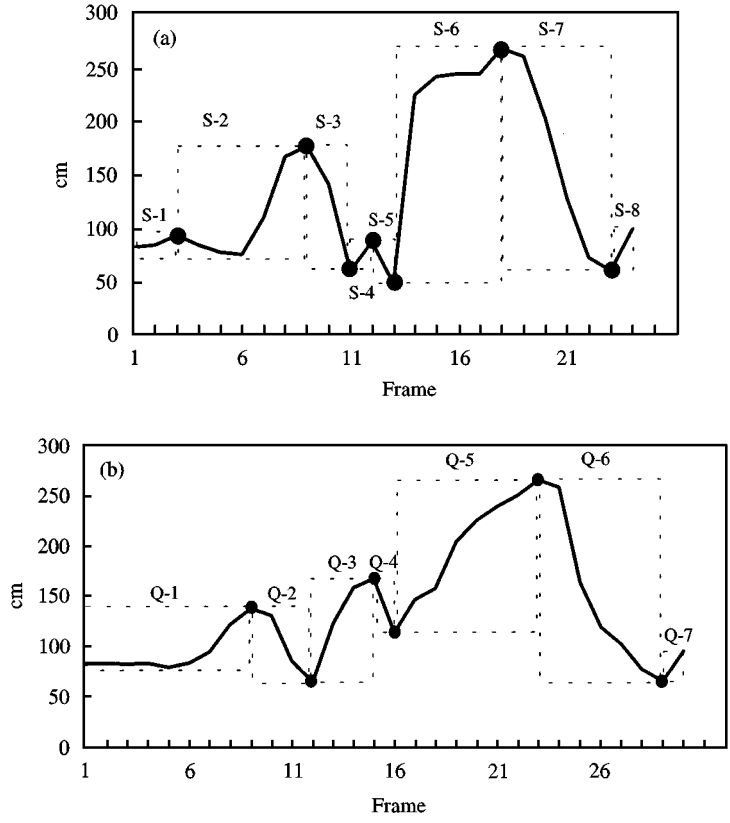


Figure 12. Frame/value diagram with bounding box and prominent feature points: (a) one sample sequence and (b) target sequence

6. System Implementation and Query Issues

6.1. Video Information System and Sport Motion Analysis Subsystem Prototype

We have implemented a video information prototype system supporting sport motion analysis on an IBM- compatible PC with MS-WINDOWS 95 using C++ language to test object extraction and to approximate the matching mechanisms (Figures 13 and 14).

The video information prototype system provides the capabilities of random access, frame stilling, frame stepping and slow play for an interactive video system [3]. The Play Back area has the functions of mark in/out logging, preview window, video file playback and display of the current processing frame. The Thumbnail Area can function as a display of six-divided frames or a set of still salient images of active video sequences or an A-to-F roll editing monitor each with a different video file or the video icons of query results. The Image Processing Area shows the results of video/image processing functions (for example, color key, caption, special effects, etc.), temporarily duplicated still frame, and region-based color feature extraction. The Single Frame Data Area

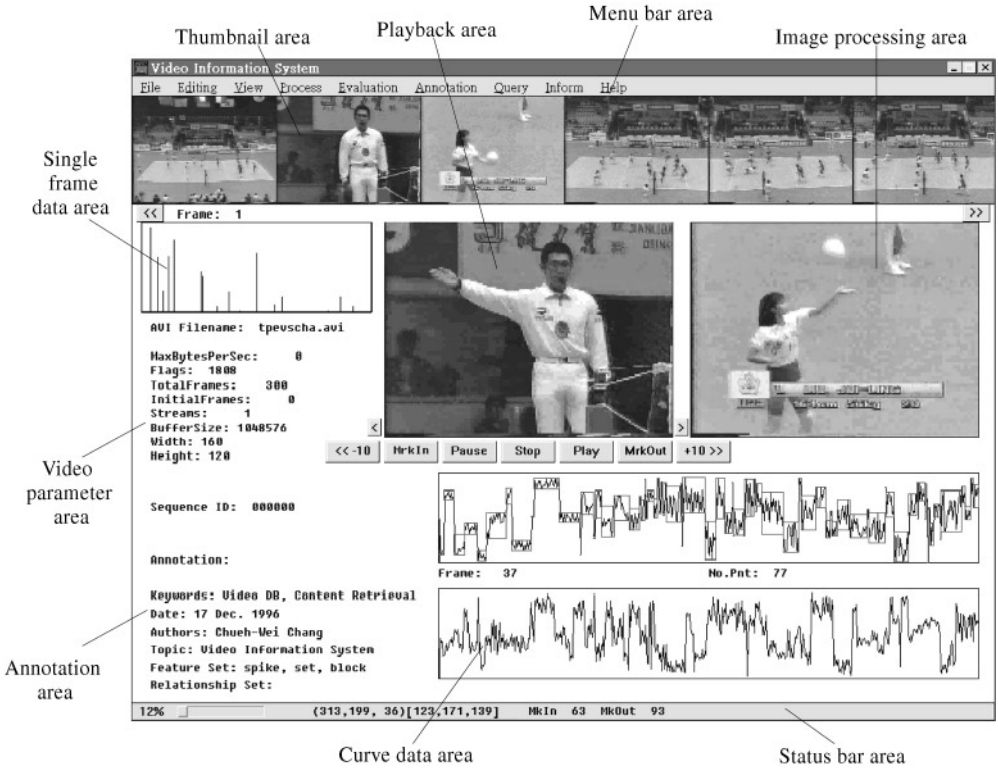


Figure 13. Video information system prototype

shows the feature values in a current processing frame (e.g. histogram, region size, etc.). The Curve Data Area shows the evaluation curve, bounding boxes, query and matched curve. The Video Parameter Area shows the related parameters of a current active video file. The Annotation Area shows the default annotation about video contents. The Status Bar Area shows the processing percentage, current cursor coordinates and corresponding R, G, B color intensities, feature value of video sequences and video editing to mark in/out cue points. The Menu Bar Area provides the functions of annotation, feature extraction, indexing, query processing and database management.

The sport motion analysis subsystem provides multiple view ports, measurement tools, measurement results display and matching mechanisms. Users can easily check the motion differences file-by-file and frame-by-frame.

6.2. Query Issues

We go back to the volleyball court and then start our queries. The spike in volleyball is one of the most difficult sport skills to perform. The most important element of good execution is proper timing. Three common errors in executing an attack are (1) beginning the approach too early, (2I) lacking height on the jump during spike and (3) contacting the ball behind the hitting shoulder. If an attacker approaches the ball too

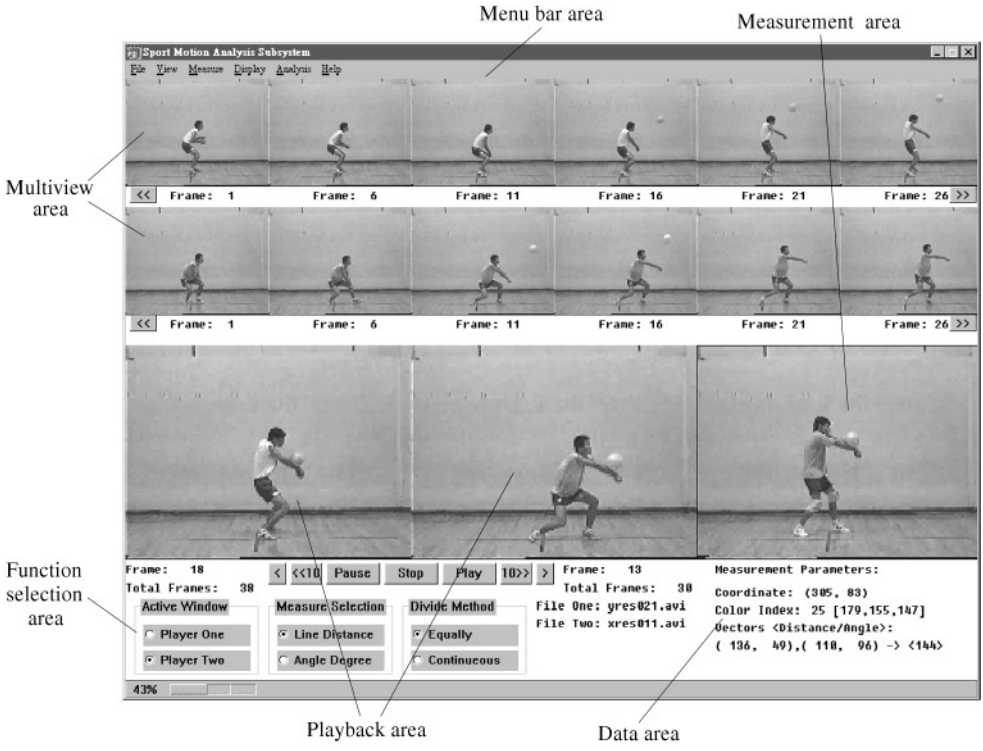


Figure 14. Sport motion analysis subsystem

early, two results are possible: (1) to stop, wait for the set and thus lose the benefit of the approach or (2) to back up to attack the ball because he has approached too far.

Therefore, the major purpose of our sport motion analysis will determine how to take a good spike from a given location on the court. According to the result of the best mapping, we can calculate the posture and timing differences, and generate a report about the mismatched posture and significant differences between the target and sample sequences. That is, by analysing the example in Section 2.2, we can observe that, in the sample sequence, the preference hand swings very fast and with large motion but, in the target sequence, the attacker seems to act slower than in the sample sequence. This result is concluded by the differences in the stable frames (frames 9–22 in the sample sequence and frames 9–28 in the target sequence), by a large overall distance 447 (the summation of sequence distance: 76, 83, 40, 179 and 69, for the respective stable subsequence groups), and by the minimum/maximum range for each alignment group. Furthermore, we can also ask questions, such as:

(1) What is one's highest spike position? We can answer this question by simply measuring the maximum altitude of the preference hand.
(2) What is one's jumping offset in this spike?—Measure the X-axis offset when jumping altitude is nonzero.

(3) How long can one stay in the air?—Measure the number of frames when jumping altitude is nonzero.
(4) What is one's average forward velocity when jumping in the air?—Measure the jumping offset divided by jumping duration.
(5) What is the minimum leg angle when one starts to jump?—Measure the leg position and calculate the leg angle manually.

Some of the answers can be obtained automatically from the system, such as (1)–(4) and some of them can only be obtained manually by inspecting the video sequences frame-by-frame.

7. Conclusions

In this paper, we attempted to set up a sport motion analysis environment with the help of digital video and personal computer and tried to maximize performance and enjoyment of the sport in the shortest possible time. By providing content-based retrieval, applications of digital video are broad in many aspects. Video records the changes of scenes according to time. Changes in video objects are quite useful for dynamic scene and motion analysis. Change of objects between different frames provides much information about the behavior of these objects in the video.

Video object segmentation, time-series data-matching and indexing are the essential preliminary steps in most video information systems. The methods discussed in this paper are representative of techniques commonly used in practice. We have successfully analysed several sport motion examples using the developed system. In addition, the techniques used for object feature extraction, curve segmentation and matching appear general in nature. Thus, we are hopeful that suitable modifications will support the system to analyse more complex video, such as color video sequences of simplistic outdoor sport scenes.

The spatio-temporal matching algorithm can be extended to several useful application domains, such as gesture recognition [23]. We only need to record the spatio-temporal feature values, then the recognition processing can be replaced by a searching method with an error tolerance. This approach will be faster and more flexible than traditional ways. We perceive this gesture recognition project as our future work.

Notation

$f(i, j, k)$	intensity function of video frames k at coordinate (i, j)
$d_{k,k+1}(i, j)$	bilevel frame difference between video frames k and $k+1$ at coordinate (i, j)
η	threshold value of frame difference
x and y	video frame resolution in X- and Y-axis, respectively
R	unit distance in centimeter (distance between two pixels)
T	number of pixels in image coordinate system
C	distance in centimeter of world coordinate system
A	target sequence

$a_1 a_2 \ldots a_n$	subsequence in the target sequence
B	sample sequence
$b_1 b_2 \ldots b_m$	subsequence in the sample sequence
p and q	shift alignment points
τ	similarity threshold
$\sim$	similarity relation
$F:[A'] \rightarrow [B']$	mapping function from subsequence A' to subsequence B'
m' and n'	the length of subsequence A' and B', respectively
$S_F(A', B')$	subsequence distance of two subsequences A' and B'
$D(A', B')$	best matched distance of two subsequences A' and B'
$\perp$	skip symbol
$M(A, B)$	overall distance of two sequences A and B
g	the number of alignment subsequence group

References

1. R. J. Beichner (1996) The impact of video motion analysis on kinematics graph interpretation skills. *American Journal of Physics* 64, 1272–1277.
2. J. Weng, T. S. Huang & N. Ahuja (1993) *Motion and Structure from Image Sequences.* Springer, Berlin.
3. D. Zollman & R. Fuller (1994) Teaching and learning physics with interactive video. *Physics Today* 47, 270–274.
4. E. Oomoto & K. Tanaka (1993) OVID: design and implementation of a video-object database system. *IEEE Transactions on Knowledge and Data Engineering* 5, 629–643.
5. S. W. Smoliar & H. Zhang (1994) Content-based video indexing and retrieval. *IEEE Multimedia* 1, 62–72.
6. M. Flickner, R. Barber, W. Cody, W. Equitz, E. Glasman, W. Niblack & D. Petkovic (1995) Query by image and video content: the QBIC system. *IEEE Computer* 28, 23–32.
7. S. Adah, K. S. Candan, S. H. Chen, K. Erol & V. S. Subrahmanian (1996) The advanced video information system: data structures and query processing. *Multimedia Systems* 4, 172–186.
8. L. A. Row, J. S. Boreczky & C. A. Eads (1995) Indexes for user access to large video databases. *IS&T/SPIE Symposium on Electrical Imaging Science and Technology*, San Jose, CA. The International Society for Optical Engineering, The Society for Imaging Science and Technology, Springfield, Virginia. pp. 1–10.
9. C. W. Chang, K. F. Lin & S. Y. Lee (1995) The characteristics of digital video and considerations of designing video databases. In: *Proceedings of the 4th International Conference on Information and Knowledge Management*, Baltimore, MD. The Association for Computing Machinery, New York. pp. 370–377.
10. C. W. Chang & S. Y. Lee (1995) Statistical and topological feature extraction and matching in video sequences. In: *National Computer Symposium*, Chung-Li, Taiwan, pp. 693–700.
11. C. W. Chang & S. Y. Lee (1996) Indexing and approximate matching for content-based time-series data in video database. In: *Proceedings of the 1st International Conference on Visual Information Systems*, Melbourne, Australia. Victoria Univ. of Technology, Melbourne City. pp. 567–576.
12. J. K. Wu, A. D. Narasimhalu, B. M. Mehtre, C. P. Lam & Y. J. Gao (1995) CORE: a content-based retrieval engine for multimedia information systems. *Multimedia Systems* 3, 25–41.
13. C. Frohlich (1986) Resource letter PS-1: physics of sports. *American Journal of Physics* 54, 590–593.
14. R. Beichner, M. DeMarco, D. Ettestad & E. Gleason (1989) VideoGraph: a new way to study kinematics. In: *Computers in Physics Instruction* (E. Redish & J. Risley, eds), Addison-Wisley, Raleigh, NC, pp. 244–245.
15. B. Bascle, P. Bouthemy, R. Deriche & F. Meyer (1994) Tracking complex primitives in an image sequence. In: *IEEE 12th International Conference on Pattern Recognition*, Jerusalem, Israel, pp. 426–431.

16. D. Koller, J. Weber, T. Huang & S. Russel (1994) Towards robust automatic traffic scene analysis in real-time. In: *IEEE 12th International Conference on Pattern Recognition*, Jerusalem, Israel, pp. 126–131.
17. K. R. Castleman (1996) *Digital Image Processing*. Prentice-Hall, Englewood Cliffs, NJ.
18. J. Aoe (1994) *Computer Algorithms: String Pattern Matching Strategies*. IEEE Computer Society Press, Los Alamos, CA.
19. D. E. Knuth, J. H. Morris & V. R. Pratt (1977) Fast pattern matching in strings. *SIAM Journal of Computing* 6, 323–350.
20. Y. P. Wang & T. Pavlidis (1990) Optimal correspondence of string subsequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12, 1080–1087.
21. C. Faloutsos, M. Ranganathan & Y. Manolopoulos (1994) Fast subsequence matching in time-series databases. In: *ACM SIGMOD*, Minneapolis, MM. The Association for Computing Machinery, New York. pp. 419–429.
22. D. Comer (1979) The ubiquitous B-tree. *ACM Computing Surveys* 11, 121–137.
23. P. Kelly & S. Moezzi (1995) Project reports: visual computing laboratory. *IEEE Multimedia* 94–99.