비디오 데이타베이스에서 이동 객체를 위한 k-워핑 알고리즘 기반 유사 부분궤적 검색

(Similar Sub-Trajectory Retrieval based on k-warping Algorithm for Moving Objects in Video Databases)

심 춘 보 † 장 재 우 ††
(Choon-Bo Shim) (Jae-Woo Chang)

요 약 이동 객체(moving objects)의 궤적(trajectories)은 내용 기반 비디오 검색을 위해 비디오의 내용이나 의미를 색인하는 데 있어 매우 중요한 역할을 한다. 따라서 본 논문에서는 비디오 데이터가 지니는 이동 객체의 궤적(moving objects' trajectories)에 대한 효율적인 검색을 위해 k-워핑(k-warping) 알고리즘에 기반한 유사 부분궤적 검색(similar sub-trajectory retrieval) 기법을 제안한다. 제안하는 방법은 궤적을 구성하는 움직임 요소 모두에 대해서 고정된 값(k)만큼까지의 반복을 허용하는 고정 반복 유사 부분궤적 검색(Fixed-Replication similar Sub-trajectories Retrieval: FRSR)과 움직임 요소 각각에 대해서 서로 다른 값으로 할당하고 그 값만큼까지의 반복을 허용하는 가변 반복 유사 부분궤적 검색(Variable-Replication similar Sub-trajectory Retrieval: VRSR) 방법이다. 제안하는 방법은 이동 객체의 궤적을 모델링하기 위해 주로 사용되는 방향만의 단일 속성(property) 뿐만 아니라, 방향, 거리, 그리고 시간 등을 포함하는 다중 속성(multiple properties)을 지원한다. 마지막으로, 성능 평가를 통해, 제안하는 k-워핑 알고리즘에 기반한 유사 부분궤적 검색 기법이 동등한 재현율을 유지하면서, 기존의 Li의 방법(no-warping)과 Shan의 OCMR방법(infinite-warping)에 비해 정확율 측면에서 좋은 성능을 보인다.

키워드 : 비디오 데이터, 이동 객체, k-워핑 알고리즘, 유사 부분궤적 검색

Abstract Moving objects' trajectories play an important role in indexing video data on their content and semantics for content-based video retrieval. In this paper, we propose new similar sub-trajectory retrieval schemes based on k-warping algorithm for efficient retrieval on moving objects' trajectories in video data. The proposed schemes are fixed-replication similar sub-trajectory retrieval(FRSR) and variable-replication similar sub-trajectory retrieval(VRSR). The former can replicate motions with a fixed number for all motions being composed of the trajectory. The latter can replicate motions with a variable number. Our schemes support multiple properties including direction, distance, and time interval as well as a single property of direction, which is mainly used for modeling moving objects' trajectories. Finally, we show from our experiment that our schemes outperform Li's scheme(no-warping) and Shan's scheme(infinite-warping) in terms of precision and recall measures.

Key words : video data, moving object, k-warping algorithm, similar sub-trajectory retrieval

1. 서 론

PCS, PDA와 같은 이동기기 보급의 확산, GPS(Global Positioning System) 활용도의 급증, 무선 데이터 통신(WDC) 및 무선 인터넷의 기술이 발달함에 따라 이동 객체의 위치 정보와 같은 다양한 정보를 수집하기가 매우 용이해졌다. 수집된 이동 객체의 위치 정보 및 이동 정보의 효율적인 관리 및 저장을 위한 연구가 최근 들어 활발히 이루어지고 있으며 특히 이러한 연구는 지리 정보 시스템(GIS)나 시공간 데이타베이스 분야에

† 비 회 원 : 전북대학교 컴퓨터공학과
cbsim@dblab.chonbuk.ac.kr
†† 종신회원 : 전북대학교 전자정보공학부 교수
전북대학교공학연구원 공업기술연구센터 연구원
jwchang@dblab.chonbuk.ac.kr
논문접수 : 2002년 6월 12일
심사완료 : 2002년 11월 18일

서 중요한 연구 대상이다[1,2,3]. 시공간 데이타베이스 분야에서의 이동 객체(Moving Object)에 대한 대표적인 질의는 다음과 같다. "*사고지점에서 가장 가까운 지역에 있는 배나 비행기를 찾아라.*" 위의 질의에서와 같이 주어진 질의와 가장 가까운 이동 객체를 찾는 최근접 질의는 시간에 따라 그 위치가 변함에 따라 공간적인 요소와 시간적인 요소를 모두 고려해서 처리할 수 있어야 한다. 한편, 비디오 데이타가 가지는 중요한 특징 중에 하나는 이동 객체에 대한 궤적(trajectory) 정보이다. 이러한 궤적 데이타는 객체의 공간적인 속성과 시간적인 속성이 결합된 시공간 관계성을 통해 표현되며, 비디오 데이타베이스 분야에서 비디오 데이타에 대한 사용자의 내용-기반 검색을 수행하는 데 있어 매우 중요한 역할을 한다[4,5]. 비디오 데이타베이스 분야에서 이동 객체에 대한 대표적인 질의는 다음과 같다. "*사용자에 의해 스케치된 이동 객체의 궤적과 유사한 궤적을 포함하는 모든 이동 객체들을 찾아라.*"

시간의 흐름에 따라 공간적 위치가 연속적으로 변하는 객체를 이동 객체(moving objects)라 하며, 이러한 이동 객체의 연속적인 움직임들의 모임을 궤적(trajectories)이라 한다. 이러한 이동객체의 궤적은 시공간 데이타베이스나 비디오 데이타베이스에서 사용자의 주된 관심의 대상이며, 내용 기반 검색을 수행하는 데 있어 매우 중요한 역할을 수행한다. 그리고, 주어진 사용자 질의 궤적과 유사한 궤적을 포함하는 이동 객체의 궤적을 찾는 것을 유사 부분궤적 검색(similar sub-trajectory retrieval)이라 한다. 유사 부분궤적 검색을 지원하기 위해서는 질의 궤적과 주어진 허용치 범위 내에서 유사한 데이타 궤적을 검색할 수 있는 근사 매칭(approximate matching)을 지원해야 한다.

이를 위해, 본 논문에서는 비디오 데이타가 지니는 이동 객체의 궤적(moving objects' trajectories)에 대한 효율적인 검색을 위해 k-워핑(k-warping) 알고리즘 기반 유사 부분궤적 검색(similar sub-trajectory retrieval) 기법을 제안한다. 제안하는 기법은 궤적을 구성하는 움직임 요소 모두에 대해서 고정된 값(k)만큼까지의 반복을 허용하는 고정 반복 유사 부분궤적 검색(fixed-replication similar sub-trajectory retrieval : FRSR)과 움직임 요소 각각에 대해서 서로 다른 값으로 할당하고 그 값만큼까지의 반복을 허용하는 가변 반복 유사 부분궤적 검색(variable-replication similar sub-trajectory retrieval : VRSR)이다. 제안하는 방법은 이동 객체의 궤적을 모델링하기 위해 주로 사용되는 방향만의 단일 속성(property)뿐만 아니라, 방향, 거리, 그리고 시간 간격 등을 포함하는 다중 속성(multiple properties)을 지원한다.

본 논문의 구성은 다음과 같다. 2장에서는 유사 부분궤적 검색에 대한 기존의 관련 연구를 살펴보고, 3장에서는 이동 객체를 위한 k-워핑 알고리즘 기반 유사 부분궤적 검색 기법을 제안한다. 4장에서는 제안하는 기법의 성능 평가를 통해, 기존의 타 연구들과의 성능 비교를 수행한다. 마지막으로, 5장에서는 결론 및 향후 연구 방향을 제시한다.

2. 관련 연구

2.1 유사 부분궤적 검색

이동 객체에 대한 연구는 다양한 분야에서 활발한 연구가 이루어지고 있다. 그러나 이동 객체의 궤적을 기반으로 주어진 질의 궤적과 데이타 궤적 사이의 유사성을 측정하여, 유사한 부분궤적을 포함하는 데이타 궤적을 검색하는 방법에 대한 연구는 Li의 방법과 Shan의 방법이 있다.

첫째, John Z. Li(이하 Li)[6,7]은 어떤 일정시간 동안 객체의 위치가 변하는 객체를 이동 객체(moving object)로 간주하고 이에 대해서 8개의 방향 즉, North (NT), NorthWest(NW), West(WT), South West (SW), South(ST), SouthEast(SE), East(ET), North East(NE)를 고려하여 이동 객체의 궤적을 표현하고 있다. 임의의 시간 간격 I_i 동안 이동 객체 A의 움직임(motion)은 (S_i, d_i, I_i)로 표현하며, 여기에서, S_i는 이동 객체 A의 변위(displacement)이고 di는 이동 객체 A의 움직임 방향(direction)을 의미한다. 따라서 어떤 일련의 주어진 시간 간격 <I_1, I_2, ...,I_n>에 대해서 이동 객체 A의 궤적은 다음의 일련의 움직임들로 표현된다.

<(S_1, d_1, I_1), (S_2, d_2, I_2), . . . , (S_n, d_n, I_n)>

Li에서는 표 1과 같이 위상 관계에 대한 유사성 거리(similarity distance)와 방향 관계에 대한 유사성 거리를 이용하여 이동 객체 A의 궤적과 이동 객체 A와 B 사이의 시공간 관계성에 대한 유사성(similarity)을 측정한다. 이동 객체 A의 궤적과 B의 궤적 사이의 유사성 측정 함수는 TrajSim(A, B)는 다음과 같다.

$$minDiff(A,B) = MIN \sum_{i=1}^{m} distance(M_i, N_{i+j})$$

$$(\forall j \quad 0 \le j \le n-m)$$

$$TrajSim(A,B) = \frac{maxDiff(A,B) - minDiff(A,B)}{maxDiff(A,B)}$$

둘째, Shan[8]은 내용 기반 비디오 검색을 위해 이동 객체의 궤적을 이루는 각각의 움직임 요소들을 단일 속

표 1 방향과 위상 관계에 대한 유사성 거리

(a) 방향 관계에 대한 유사성 거리

	NT	NW	NE	WT	SW	ET	SE	ST
NT	0	1	1	2	3	2	3	4
NW	1	0	2	1	2	3	4	3
NE	1	2	0	3	4	1	2	3
WT	2	1	3	0	1	4	3	2
SW	3	2	4	1	0	3	2	1
ET	2	3	1	4	3	0	1	2
SE	3	4	2	3	2	1	0	1
ST	4	3	3	2	1	2	1	0

(b) 위상 관계에 대한 유사성 거리

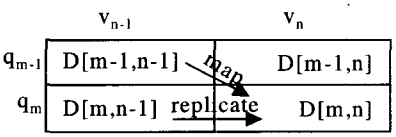
	DJ	TC	EQ	IN	CB	CT	CV	OL
DJ	0	1	6	4	5	4	5	4
TC	1	0	5	5	4	5	4	3
EQ	6	5	0	4	3	4	3	6
IN	4	5	4	0	1	6	7	4
CB	5	4	3	1	0	7	6	3
CT	4	5	4	6	7	0	1	4
CV	5	4	3	7	6	1	0	3
OL	4	3	6	4	3	4	3	0

성인 방향을 위해 실제 각도(0~360)를 이용하여 표현하고, 사용자 질의 궤적 $Q=(q_1,q_2, ...,q_M)$와 유사한 부분 궤적을 포함하는 데이타 궤적 $V=(v_1,v_2, ...,v_N)$사이의 유사성을 측정하기 위해 OCM(Optimal Consecutive Mapping)과 OCMR(Optimal Consecutive Mapping with Replication)의 두 가지 유사성 측정 알고리즘을 제안한다. 먼저, OCM 알고리즘은 정확 매칭(exact matching)으로 질의 궤적과 데이타 궤적사이의 각 움직임 요소들 간에 일대일 매핑으로 유사성을 측정한다. 즉, 질의 궤적 Q를 이루는 움직임 요소들을 데이타 궤적 V의 첫 번째 움직임 요소(v_1)으로부터 질의 궤적의 움직임 수(M)만큼 순서대로 매핑하여 유사성을 측정한다. 마찬가지로, 데이타 궤적의 두 번째 움직임 요소(v_2)으로부터 M만큼 순서대로 매핑해서 유사성을 측정한다. 이렇게 구한 유사성들 중에서 가장 작은 값을 질의 궤적 Q와 데이타 궤적 V사이의 유사성으로 선택한다. OCMR 알고리즘은 근사 매칭을 지원하며, 그림 1에서와 같이 사용자로부터 주어진 질의 궤적을 이루는 각각의 움직임 요소들(q_i)과 데이타 궤적을 이루는 각각의 움직임 요소들(v_j)사이에 유사성을 계산하는 데 있어, 질의 움직임 요소들 중에서 자기 자신을 반복시킨 것(replicate)과 그렇지 않은 것(map)을 비교해서 더 작은 것을 선택한다. 여기서 $d(q_i, v_j)$는 q_i와 v_j사이의 거리 함수(distance function)이고, D[M,N]는 질의 궤적과 데이타 궤적사이의 반복(replication)을 이용한 최소 거리(minimum distance)를 구하기 위한 테이블이다. 그림 2는 근사 매칭을 위한 OCMR알고리즘을 나타낸다.

2.2 유사 서브시퀀스 검색

유사 서브시퀀스(subsequence) 검색[9,10,11]은 주어진 질의 시퀀스를 포함하는 유사한 데이타 시퀀스를 검색하는 것으로 주가 데이타, 상품 판매량, 날씨 데이타, 의료 데이타와 같은 응용 분야에서 많은 연구가 이루어지고 있다. 우선, 시퀀스 데이타베이스는 다양한 길이의 시퀀스들로 구성되며, 시퀀스 $S(=<s[1], s[2], ..., s[|S|]>)$는 일정한 시간 주기마다 얻어진 연속된 실수 값들로 이루어진다. 여기서, |S|는 시퀀스의 길이이다. s[i]는 S의 i번째 요소를 나타내며, s[i:j]는 i번째 위치에서 j번째 위치를 포함하는 서브시퀀스를 의미한다. s[i:-]는 i번째 위치에서 시작해서 시퀀스 S의 마지막 요소까지를 나타낸다. ()는 요소가 없는 널 시퀀스(null sequence)를 의미한다.

주어진 질의 시퀀스와 데이타 시퀀스를 구성하는 각



$$D[m,n]=\min\begin{cases} D[m-1,n-1]+d(q_m, v_n) \\ D[m-1,n]+d(q_m, v_n) \end{cases}$$

그림 1 D[m,n]과 D[i,j]사이의 OCMR의 관계

```
Algorithm Optimal Consecutive Mapping with
Replication(OCMR)

for j=0 to N-M do D[0,j]=0;
for i=1 to M-1 do D[i,i-1]=∞;
for j=0 to N-M do
   for i=1 to M-1 do
      D[i,i+j]=min(D[i-1,i+j-1]+d(qi,vi+j),
      D[i,i+j-1]+d(qi,vi+j));

D[M,M]=D[M-1,M-1]+d(qM,vM);

for j=M+1 to N do
   D[M,j]=min(D[M-1,j-1]+d(qM,vj),D[M,j-1]);

return D[M,N];
```

그림 2 근사 매칭을 위한 OCMR 알고리즘

요소들간의 유사성을 측정하기 위해서 정의 1과 같은 거리 함수가 요구된다.

[정의 1] 길이가 n인 두 시퀀스 S와 Q 사이의 유사한 정도를 계산하기 위한 거리 함수 L_p는 다음과 같이 정의된다. L_1은 맨하탄 거리(manhatann distance). L_2는 유클리드 거리(euclidean distance), L_∞는 대응되는 각 쌍의 거리 중 최대 거리를 의미한다.

$$L_p(S,Q) = \left(\sum_{i=1}^{n} |s[i]-q[i]|^p\right)^{\frac{1}{p}}, 1 \le p \le \infty$$

효율적인 유사 서브시퀀스 검색을 위해 정규화(normalization), 이동 평균(moving average), 타임 워핑(time warping)등의 다양한 알고리즘들이 제안되었다. 그 중에서도 특히 타임 워핑 기법은 시퀀스내의 각 요소 값을 임의의 수만큼 반복시키는 것을 허용하는 알고리즘이다. 이를 통해 사용자의 정확하지 못한 대략적인 질의에 대해서 어느 정도의 허용치 범위내에서 사용자의 질의를 변형함으로써 사용자가 원하는 검색 결과를 보장하는 근사 매칭을 지원한다. 다음은 유사 서브시퀀스 검색을 위해 제안된 타임 워핑 거리를 나타내는 정의이다.

[정의 2] 두 시퀀스 S와 Q 사이의 타임 워핑 거리 D_{tw}는 다음과 같이 재귀적으로 정의된다. D_{base}는 기본 거리 함수로서 L_p 중 응용에 적합한 것을 선택하여 사용한다.

$D_{tw}((),()) = 0$

$D_{tw}(S,()) = D_{tw}((),Q) = \infty$

$D_{tw}(S,Q) = D_{base}(S[1], Q[1]) + \min(D_{tw}(S,Q[2:-]), D_{tw}(S[2:-],Q), D_{tw}(S[2:-],Q[2:-]))$

$D_{base}(a, b) = |a - b|$

타임 워핑 변환을 이용한 유사 서브시퀀스 검색은 정의 2에서와 같이 질의 시퀀스와 데이타 시퀀스를 구성하는 임의의 요소에 대해서 임의의 수만큼 제한 없이 반복시키는 것을 허용한다. 정의 2의 타임 워핑 거리를 알고리즘으로 표현하면 그림 3과 같다.

3. k-워핑 알고리즘 기반 유사 부분궤적 검색

3.1 k-워핑 알고리즘

본 논문에서는 비디오 데이타베이스에서 이동 객체의 유사 부분궤적 검색을 지원하고 효율적인 근사 매칭(approximate matching)을 위해 다음의 세 가지 고려사항을 제시한다.

고려사항 1 : 시퀀스 데이타베이스에서 유사 서브시퀀스 검색을 위해 사용되었던 타임 워핑 변환은 질의 시퀀스 뿐만 아니라 데이타 시퀀스에 대해서도 임의의 요소가 무한히 반복되는 것을 허용한다. 그러나, 비디오 데이타베이스에서의 유사 부분궤적 검색을 위해서는 질의 궤적을 구성하는 움직임 요소에 대해서만 반복을 허

```
float TimeWarping_Distance(int num1, float *vals1, int num2, float *vals2)
{
  Input:
       num1 : the number of Data Sequence;
       num2 : the number of Query Sequence;
       vals1 : Data Sequence;
       vals2 : Query Sequence;
  Output:
       dtable[num1][num2]; time-warping distance;

       float dtable[num1][num2];
       dtable[0][0] = fabs(vals1[0] - vals2[0]);
       for (int j = 1; j < num2; j++)
          dtable[0][j] = fabs(vals1[0] - vals2[j]) + dtable[0][j-1];

       for (int i = 1; i < num1; i++)
          dtable[i][0] = fabs(vals1[i] - vals2[0]) + dtable[i-1][0];

       for (int i = 1; i < num1; i++)
          for (int j = 1; j < num2; j++)
             dtable[i][j] = fabs(vals1[i] - vals2[j]) +
                    fmin3 (dtable[i][j-1], dtable[i-1][j], dtable[i-1][j-1]);

       return dtable[num1-1][num2-1];
}
```

그림 3 유사 서브시퀀스를 위한 타임 워핑 거리 알고리즘

용해야한다.

고려사항 2 : 시퀀스 데이타베이스에서 유사 서브시퀀스 검색을 위해 사용되었던 타임 워핑 변환은 임의의 요소들을 무한히 반복하는 것을 허용한다. 그에 반해, 유사 부분궤적 검색을 위해서는 무한 반복(infinite warping)보다는 임의의 고정된 수(k)만큼까지만 반복을 허용해야 한다. 단, 반복의 횟수는 유사 부분궤적을 적용하는 응용 분야에 따라 제한될 수 있다.

고려사항 3 : 이동 객체의 궤적을 이루는 움직임 요소를 모델링하기 위해서는 방향만의 단일 속성뿐만 아니라, 방향, 거리, 그리고 시간 간격 등을 포함하는 다중 속성을 지원해야 한다.

여기서 고려사항 1은 일반적으로 유사 부분궤적 검색에서 근사 매칭을 지원하기 위함이고, 고려사항 2와 3은 고려사항 1을 이용한 근사 매칭의 효율성을 향상시키기 위함이다. 아울러, 고려사항 2와 3은 유사 부분궤적 검색을 적용하려는 응용 분야가 스포츠 비디오 데이타 검색 분야의 경우(예> 축구, 농구, 하키 등) 매우 중요한 사항이다.

기존의 타임 워핑 변환을 이용한 유사 서브시퀀스 검색 기법은 위의 세 가지 고려사항 모두를 만족시키지 않는다. 이는 시퀀스 데이타베이스에서 사용하는 시퀀스 데이타의 특성과 비디오 데이타베이스에서 사용하는 이동 객체의 궤적 데이타의 특성이 서로 상이하기 때문이다. 일반적으로, 시퀀스 데이타는 시간의 흐름에 따라 연속된 실수 값들로 구성되기 때문에 조밀조밀하고 정교한 특징을 지니며, 하나의 시퀀스를 구성하는 요소의 수는 적어도 수십 개에서 수백 개에 이른다. 반면에, 비디오 데이타베이스에서 이동 객체의 궤적 데이타 는 시간의 흐름에 따른 이동 객체의 움직임 요소들로 구성되며, 궤적을 이루는 움직임 요소의 수가 수십 개 내외이다. 아울러, 이동 객체의 궤적을 기반으로 근사 매칭을 위해 제안되었던 Shan의 OCMR 알고리즘은 고려사항 1은 만족시키지만, 고려사항 2와 3은 만족시키지 못한다.

따라서, 본 논문에서는 위에서 언급한 세 가지 고려사항을 모두 만족시키고, 질의 궤적과 주어진 허용치 범위 내에서 유사한 데이타 궤적을 검색할 수 있는 근사 매칭을 지원하는 새로운 k-워핑 알고리즘 기반 유사 부분궤적 검색 기법을 제안한다. 제안하는 기법은 기존의 시퀀스 데이타베이스에서 유사 서브시퀀스 검색을 위해 사용되었던 타임 워핑 변환 알고리즘을 이동 객체의 궤적 데이타 응용에 맞게 새로 구성한 기법이다.

한편, 본 논문에서는 고려사항 3을 만족시키기 위해, n차원의 속성으로 이루어진 움직임 요소를 가지는 이동 객체의 궤적을 정의 3과 같이 표현한다.

[정의 3] 이동 객체의 궤적 S(=<s[1], s[2], ..., s[|S|]>)는 일련의 연속된 움직임 요소들의 집합으로 표현하며, 궤적을 구성하는 임의의 움직임 요소 s[i]=(s[i,1], s[i,2], ..., s[i,n])로 나타낸다.

아울러, 이동 객체의 궤적을 2차원의 속성으로 나타내는 움직임 요소들의 집합으로 표현하며, 본 논문에서는 방향 정보를 위해서 실제 각도를 이용하여 표현하기 때문에 방향을 각도로 표현한다. 이는 정의 4와 같이 나타낸다.

[정의 4] 이동 객체의 궤적 S(=<s[1], s[2], ..., s[|S|]>)는 일련의 연속된 움직임 요소들의 집합으로 표현하며, 임의의 움직임 요소들은 각각 2차원 속성 즉, 각도(angle)와 거리(distance)로 표현한다. 즉, 움직임 궤적을 구성하는 움직임 요소 s[i]는 (A_i, D_i)로 나타낸다.

주어진 사용자 질의 궤적의 하나의 움직임 요소와 데이타 궤적의 하나의 움직임 요소 사이의 유사성을 측정하기 위해, 정의 5와 같은 거리 함수 d_{df}를 정의한다.

[정의 5] 이동 객체의 궤적 S를 구성하는 임의의 움직임 요소 s[i]와 질의 궤적 Q를 구성하는 임의의 움직임 요소 q[j]사이의 거리 함수(distance function) d_{df}(s[i],q[j])는 다음과 같이 정의된다. 여기서 d_{ang}는 움직임 요소의 각도간의 거리 함수를 나타내고, d_{dis}는 움직임 요소의 거리간의 거리함수를 의미한다. 그리고 s[i,1]와 s[i,2]는 각각 궤적 S의 i번째 움직임 요소를 구성하는 각도와 거리를 나타낸다. α와 β는 각각 각도와 거리를 위한 가중치를 의미한다.(α+β=1.0)

if| s[i,1]-q[j,1]| > 180 Then

d_{ang}(s[i,1],q[j,1]) = (360 - |s[i,1]-q[j,1]|)

else

d_{ang}(s[i,1],q[j,1]) =| s[i,1]-q[j,1] |

d_{dis} = | s[i,2] - q[j,2] |

$$d_{df} = ((d_{ang}/180) * \alpha) + ((d_{dis}/100) * \beta)$$

고려사항 1과 2를 만족시키면서 효율적인 유사 부분궤적 검색을 위해 기존의 유사 서브시퀀스 검색에서 사용되었던 정의 2를 변형하여 정의 6과 같은 새로운 k-워핑 거리 함수를 정의한다.

[정의 6] 두 궤적 S와 Q 사이의 k-워핑 거리 D_{kw}는 다음과 같이 재귀적으로 정의된다.

$$D_{kw}(0,0) = 0$$

$$D_{kw}(S,0) = D_{kw}(0,Q) = \infty$$

$$D_{kw}(S,Q) = D_{base}(S[1],Q[1]) + \min(\{D_{kw}(S[$$

2+i:-],Q),0≤i<k}, D_{kw}(S[2:-],Q[2:-]))

$D_{base}(a, b) = d_{df}(a, b)$

KW_CM_Table() 루틴은 고려사항 1을 만족시키기 위한 함수로, 기존의 시퀀스 데이타베이스에서 사용되었던 타임 워핑 거리를 계산하기 위한 알고리즘을 변형하여, 주어진 사용자 질의 궤적 Q를 구성하는 임의의 움직임 요소들에 대해서만 무한히 반복해서 데이타 궤적과 질의 궤적 사이에 거리의 차들이 누적된 테이블(kwTbl)을 만든다(그림 4). 여기서 누적 테이블의 마지막 행의 마지막 열의 값 즉, kwTbl[|S|-1][|Q|-1]은 질의 궤적을 구성하는 임의의 움직임 요소에 대해서만

```
void KW_CM_Table(S, Q)
{
  Input:
      S: Data Trajectory, Q: Query Trajectory;
  Output:
      kwTbl[][]: cumulative distance table;

   kwTbl[0][0] = ddf(S[0], Q[0]);
   for j=1 to |Q|-1 do
      kwTbl[j-1][j] = ∞;

   for i=1 to |S|-1 do
      kwTbl[i][0] = ddf(S[0],Q[0]) + kwTbl[i-1][0];

   for i=1 to |Q|-1 do
      for j=i to |S|-1 do
         kwTbl[i][j] = ddf(S[i],Q[j])+ min(kwTbl[i-1][j],kwTbl[i-1][j-1]);
}
```

그림 4 두 궤적사이의 누적 테이블 생성 함수

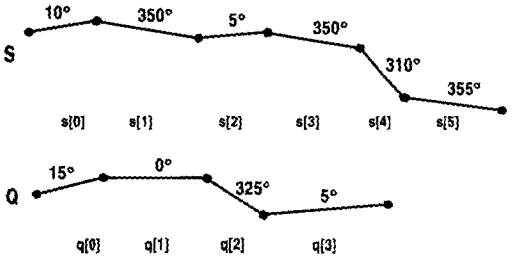
```
int k-Warping_Distance(S, Q, kwTbl, KW_List)
{
  Input:
      S: Data Trajectory, Q: Query Trajectory;
      kwTbl[][]: cumulative distance table;
      KW_List[] : the number of warping(replication) for each motion;
  Output:
      minDist: minimum distance acquired using k-warping

   lo_wp_cnt=k;              //local warping count
   gl_wp_cnt=|S|-|Q|;  //global warping count
   i=|Q|-1; j=|S|-1;
   minDist = ddf(S[j], Q[i]);
   while(1) {
      if( i==0 && j==0) break;
      else {
         if( (i>0) && (kwTbl[i][j-1] < kwTbl[i-1[j-1]) && lo_wp_cnt>0) {
               j--; lo_wp_cnt--;          gl_wp_cnt--;
               minDist += ddf(S[j],Q[i]);
         }
         else if( (i*k<gl_wp_cnt) && (lo_wp_cnt>0) && (gl_wp_cnt!=0) ) {
               j--; lo_wp_cnt--; gl_wp_cnt--;
               minDist += ddf(S[j],Q[i]);
         }
         else {
               i--; j--; lo_wp_cnt = k;
               minDist += ddf(S[j],Q[i]);
         } // end of if
      } // end of if
   } // end of while
   return minDist;
}
```

그림 5 k-워핑 거리 측정 알고리즘

무한 반복을 허용하면서 데이타 궤적 S와 질의 궤적 Q 사이의 가장 작은 값을 나타낸다.

k-Warping_Distance()루틴은 고려사항 2를 만족시키기 위한 것으로 KW_CM_Table() 함수에서 구해진 누적 테이블을 기반으로 질의 궤적 Q를 구성하는 임의의 움직임 요소들에 대해서 반복(replication) 횟수를 최대한 k번까지 허용함으로써 질의 궤적 Q와 데이타 궤적 S사이의 k-워핑 거리를 계산한다. 그림 5는 본 논문에서 제안하는 k-워핑 거리를 구하는 알고리즘이다. 이는 알고리즘 특성상 데이타 궤적의 움직임 요소를 행(row)으로 표현하고 질의 궤적의 움직임 요소를 열(column)로 표현할 때, 첫 번째 행의 첫 번째 열의 요소에서부터 시작해서 마지막 행의 마지막 열의 요소까지 단지 질의 궤적의 움직임 요소들에 대해서만 최대한 k번까지 반복을 허용하면서 가장 짧은 거리를 찾아가는 알고리즘이다. 이를 위해, 본 논문에서는 bottom-up방식 대신 top-down방식으로 찾아가는 방식을 선택한다. 이유는 bottom-up 방식과 같은 경우 첫 번째 행의 첫 번째 열의 요소에서부터 시작해서 마지막 행의 마지막 열의 요소까지 단지 질의 궤적의 움직임 요소들에 대해서만 최대한 k번까지 반복을 허용하면서 가장 짧은 거리를 찾고자 할 때, 마지막 행의 마지막 열의 요소까지 도달하지 못하는 문제점이 발생한다. 그림 4와 그림 5에서 데이타 궤적 S와 질의 궤적 Q를 구성하는 각각의 움직임 요소들 간의 거리는 정의 6에서 제안하는 새로운 거리 함수를 이용한다. 즉, 궤적을 구성하는 움직임 요소들은 각각 각도와 거리의 쌍으로 이루어져 있기 때문에 이 두 가지 속성을 모두 고려해서 거리를 계산한다.

예를 들어, 그림 6(a)에서와 같이 데이타 궤적 S = {10°, 350°, 5°, 350°, 310°, 355°}와 질의 궤적 Q = {15°, 0°, 325°, 5°} 사이에 유사성 거리를 그림 4와 그림 5의 k-워핑 알고리즘을 이용해서 구하면 다음과 같다. 먼저, 그림 6(b)와 같이 그림 4에서의 알고리즘을 이용하여 질의 궤적 Q을 구성하는 움직임 요소들에 대해서만 무한히 반복을 허용하여 가장 작은 거리값(minimum distance)를 구할 수 있는 누적 테이블을 생성한다. 즉, 마지막 행의 마지막 열의 값은 데이타 궤적 S와 질의 궤적 Q사이에 질의 궤적 Q에 대해서만 무한 반복을 통해 구할 수 있는 가장 작은 거리값인 '55'를 나타낸다. 여기서 q[1]의 움직임 요소는 각각 s[1], s[2], s[3]에 반복해서 대응된다. 마지막으로, 미리 구해놓은 누적 테이블을 기반으로 마지막 행의 마지막 열의 요소에서부터 시작해서 첫 번째 행의 첫 번째 열의 요소까지 질의 궤적 Q의 움직임 요소에 대해서만 주어진 k(=1)값까지만 반복을 허용하면서 가장 작은 거리 값으로 갈 수 있는 경로를 찾은 후에 그 경로에 해당하는 각 요소들 간의 거리 차의 합을 구한다. 즉,|q[0]-s[0]|+|q[0]-s[1]|+|q[1]-s[2]|+|q[1]-s[3]|+|q[2]-s[4]|+|q[3]-s[5]|= 5+25+5+10+15+10='70'이 값이 바로 k-워핑 알고리즘을 이용한 두 궤적 사이의 최소 거리값 이다. 본 논문에서는 궤적을 구성하는 각각의 움직임 요소들을 각도와 거리를 모두 고려하고 있다. 그러나 그림 6의 예제에서는 편의상 각도만을 고려해 k-워핑 알고리즘을 적용하여 두 궤적 사이의 거리를 측정한다.



(a) 예제(각도만 고려)

355	170	85	75	55
310	130	80	45	100
350	65	30	45	70
5	40	20	55	∞
350	30	15	∞	?
10	5	∞	?	?
S / Q				

(b) k-워핑 알고리즘의 예(k=1)

그림 6 k-워핑 알고리즘을 이용한 유사성 측정의 예

3.2 유사 부분궤적 검색

본 절에서는 3.1절에서 제안한 k-워핑 알고리즘을 기반으로 시스템 측면을 고려한 고정 반복-기반 유사 부분궤적 검색(FRSR) 방법과 사용자 측면을 고려한 가변 반복-기반 유사 부분궤적 검색(VRSR) 방법을 제안한다.

3.2.1 고정 반복-기반 유사 부분궤적 검색(FRSR) 방법

제안하는 고정 반복-기반 유사 부분궤적 검색(이하 FRSR) 방법은 이동 객체의 궤적을 구성하는 움직임 요소들 모두에 대해서 고정된 값(k)만큼까지만 반복을 허용한다. 즉, 찾고자 하는 데이타 궤적을 검색하기 위해, 사용자 질의 궤적을 구성하는 움직임 요소들

에 한해서만 시스템 내에서 정해진 고정된 값(k)만큼까지의 최대한 반복을 허용한다. 이를 통해, 정확하지 못한 질의 궤적에 허용치 범위 내에서 변형을 가해 처음 주어진 질의 궤적보다 검색의 효율성을 올릴 수 있다.

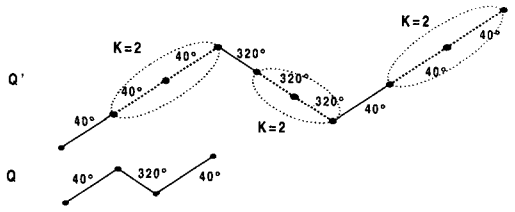


그림 7 FRSR 방법을 위해 변형된 질의 궤적(k=2)

움직임 요소에 대한 고정 반복 횟수 k는 유사 부분궤적 검색을 적용하려는 응용 분야의 데이타 도메인의 특성에 따라 유동적일 수 있다. 예를 들면, 축구, 농구, 하키등과 같은 스포츠 비디오 데이타 내의 축구공(ball)이나 하키퍽(puck)과 같은 주요 이동 객체를 위한 유사 부분궤적 검색시에는, 궤적을 이루는 움직임 개수가 적기 때문에 고정 반복 횟수 k를 낮게 정하는 것이 검색 성능을 저하시키지 않는다. 반면에, PCS나 PDA와 같은 휴대형 단말기기 추적과 같은 응용 분야의 경우는 궤적을 구성하는 움직임 요소의 개수가 많기 때문에 고정 반복 횟수 k를 높게 정하는 것이 좋다. 그림 7은 FRSR 방법을 위해 주어진 사용자 질의 궤적을 k만큼까지 반복을 허용해서 얻은 질의 궤적을 나타낸다. 그림 8은 k-워핑 알고리즘 기반 FRSR 방법을 알고리즘으로 나타낸 것이다.

3.2.2 가변 반복-기반 유사 부분궤적 검색(VRSR) 방법

가변 반복-기반 유사 부분궤적 검색(이하 VRSR) 방법은 이동 객체의 궤적을 구성하는 움직임 요소들 모두에 대해서 고정된 값(k)만큼까지의 반복을 허용하는 대신, 궤적을 구성하는 움직임 요소들 각각에 대해서 서로 다른 반복 횟수를 지정할 수 있는 방법이다. 즉, 찾고자 하는 데이타 궤적을 검색하기 위해 사용자 질의 궤적을 구성하는 움직임 요소들에 대해 사용자가 직접 각각의 움직임 요소들에 서로 다른 값만큼까지의 최대한 반복을 허용할 수 있도록 하는 방법

```
int FRSR(Q, Results)
{
  Input:
      Q: Query Trajectory;
  Output:
      Results : Retrived Results including (id, dist)
   Variable:
      k;                              // the number of warping
      t_value;          // predefined threshold value
      Si ;                    // i th data trajectory in database
      count = 0;        // the number of retrieved results
      ST[];                   // sub trajectory

      for i=0 to   |DB|-1 do {
          for j=|Q| to |Q|*(k+1) do {
              for m=0 to |Si| j do {
                  minKWDist=∞; kwTbl=[|Si|] [|Q|]; ST=[];
                  Gen_Sub_Trajectory(Si, j, m, ST);
                  KM_CM_Table(Si, Q);
                  kwDist=k_Warping_Distance(Si, Q, kwTbl, k);
                  if kwDist < minKWDist
                      minKWDist = kwDist;
              } // end of m
          } // end of j
          if minKWDist > t_value {
              Results[count].id = Si.id; Results[count].dist = minKWDist;
              count++;
          }
      } // end of for i

      return count;
}
```

그림 8 FRSR 방법의 알고리즘

이다. 이를 통해, 사용자는 질의 궤적을 구성할 때, 움직임 요소들 각각에 대해서 사용자가 검색하고자 하는 데이타 궤적의 유형에 따라 원하는 만큼의 가중치를 지정할 수 있다. 따라서 주어진 움직임 요소들의 가중치에 따라 질의 궤적 전체가 아니라, 질의 궤적 내의 사용자가 관심 있는 부분에 가중치를 지정하여 보다 효율적으로 원하는 데이타 궤적을 검색할 수 있다. 그림 9는 VRSR 방법을 위해 주어진 사용자 질의 궤적을 움직임 요소들의 각각에 대해서 서로 다른 반복 횟수를 허용해서 얻은 질의 궤적을 나타낸다. 그림 10은 k-워핑 알고리즘 기반 VRSR 방법을 알고리즘으로 나타낸 것이다.

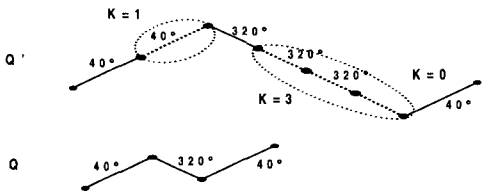


그림 9 VRSR 방법을 위한 변형된 질의 궤적(k=1,3,0)

FRSR 방법과 VRSR 방법은 모두 효율적인 유사 부분궤적 검색을 위해 제안된 k-워핑 알고리즘에 기반을 둔다. 또한, 시퀀스 데이타베이스에서 유사 서브시퀀스 검색을 위해 데이타 시퀀스와 질의 시퀀스 모두에 대해서 반복을 무한히 허용하는 타임 워핑 알고리즘과 달리, 단지 질의 궤적을 구성하는 움직임 요소들에 한해서만 반복을 허용한다. FRSR 방법은 이동 객체의 궤적을 구성하는 움직임 요소들 모두에 대해서 고정된 값(k)만큼까지만 반복을 허용하는 반면에, VRSR 방법은 궤적을 구성하는 움직임 요소들 각각에 대해서 서로 다른 반복 횟수를 지정할 수 있는 방법이다. FRSR 방법은 반복 횟수를 유사 부분궤적 검색을 적용하려는 응용 분야의 데이타 도메인의 특성에 맞춰 시스템 내에서 결정하는 반면에, VRSR 방법은 사용자가 찾고자 하는 데이타 궤적을 검색하기 위해, 주어진 질의 궤적을 구성하는 움직임 요소들에 각각에 대해서 서로 다른 값으로 반복 횟수를 결정함으로써 움직임 요소들에 대해 가중치를 지정할 수 있다.

```
int VRSR(Q, KW_List, Results)
{
  Input:
     Q: Query Trajectory;
     KW_List : k-Warping List for Query Trajectory;
  Output:
     Results : Retrieved Results including {id, dist}
  Variable:
     t_value;              // predefined threshold value
     Si ;                        // i-th data trajectory in database
     count = 0;            // the number of retrieved results
     ST[];                       // sub-trajectory

     for i=0 to   |DB|-1 do {
          for j=|Q| to |Q|*(KW_list[j]+1) do {
               for m=0 to |Si|-j do {
                    minKWDist=∞; kwTbl=[|Si|][|Q|]; ST=[];
                    Gen_Sub_Trajectory(Si, j, m, ST);
                    KM_CM_Table(Si, Q);
                    kwDist=k-Warping_Distance(Si, Q, kwTbl, KW_List);
                    if kwDist < minKWDist
                         minKWDist = kwDist;
               } // end of m
          } // end of j
          if minKWDist > t_value {
               Results[count].id = Si.id;
               Results[count].dist = minKWDist;
               count++;
          }
     } // end of for i

     return count;
}
```

그림 10 VRSR 방법의 알고리즘

따라서 FRSR 방법은 사용자의 관심 영역이 질의 궤적 전체에 있으며 시스템 측면을 고려한 방법에 비해, VRSR 방법은 사용자의 관심 영역이 질의 궤적 전체가 아니라 특정한 일부분으로 한정시킬 수 있으며 사용자의 측면을 고려한 방법이다. 표 2는 k-워핑 알고리즘 기반 FRSR 방법과 VRSR 방법의 차이점을 나타낸다.

표 2 FRSR 방법과 VRSR 방법의 차이점

	FRSR 방법	VRSR 방법
반복을 허용하는 궤적	질의 궤적	
기반 알고리즘	k-워핑 알고리즘	
움직임 요소의 반복 횟수	고정 반복	가변 반복
반복 횟수 결정	시스템	사용자
궤적 내의 사용자 관심 영역	궤적의 전체	궤적의 일부분
고려 대상	시스템 측면	사용자 측면

4. 실험 및 성능평가

제안하는 k-워핑 알고리즘 기반 유사 부분궤적 검색(이하 k-워핑 방법) 기법의 유용성을 검증하기 위해, 실제 축구 비디오 데이타로부터 축구공의 궤적을 추출하여 성능 평가를 수행한다. 축구 비디오 데이타는 사용자의 주된 관심 객체인 축구공이 경기장을 배경으로 많은 움직임 궤적을 가지는 특징이 있어 이동 객체의 움직임 궤적을 추출하는 데 매우 용이하다. 표 3은 성능평가를 위해 사용된 실험 데이타를 나타낸다.

표 3 성능 평가를 위한 실험 데이타

인 자	실험 데이타
데이타 도메인	축구 비디오 데이타
주요 움직임 객체	축구공
실험 데이타의 수	350개
궤적의 평균 움직임의 수	8개
질의 데이타의 수	40개
질의 궤적의 평균 움직임의 수	3개

실험에 사용되는 축구 비디오 데이타는 MPEG 동영상 파일 포맷(*.mpeg)으로 대부분이 "골을 넣는 장면"을 포함하고 있다. 또한 축구 비디오 데이타로부터 움직임 궤적의 추출은 수작업을 통해 이루어지는 데, 정확한 궤적 추출을 위해 카메라가 Zoom Out된 상태 즉, 축구공, 선수들, 골대 등의 객체가 전체적으로 화면상에 나타나는 샷들만을 선별하여 움직임 궤적을 추출한다. 아울러, 350개의 비디오 데이타로 구성된 축구 비디오 데이타베이스로부터 실제로 '골인'이 되는 대표적인 움직임 궤적을 기준으로 축구 경기장 중앙선(하프라인)의 오른쪽 진영(right field)과 왼쪽 진영(left field)에서 각각 20개씩 총 40개의 질의를 추출하여 사용한다. 한편, 이들 질의는 일반적인 비디오 질의로는 부적당하며, 현재 실험을 위해 축구 비디오 데이타에 한정하여 사용한다.

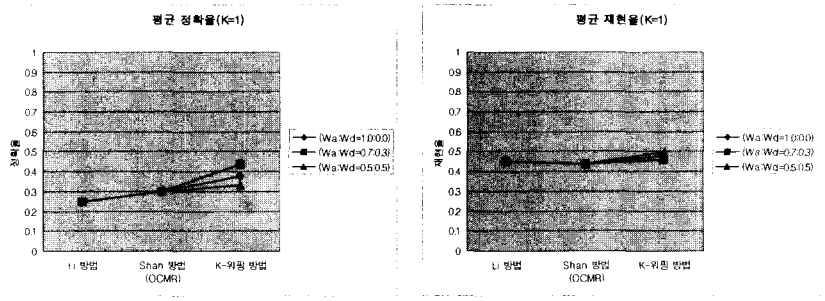
성능 평가를 위하여 128MB 메인 메모리를 갖는 펜티엄-PC 상에서 Microsoft Visual C++ 컴파일러를 사용하여 Li 방법과 Shan 방법 그리고 본 논문에서 제안하는 k-워핑 방법을 프로그래밍하여 실험을 수행하며, 검색 효과(retrieval effectiveness) 측면 즉, 정확율(precision)과 재현율(recall)[12]를 성능 평가 지수로 사용한다. 먼저, RVQ(Relevant Video data to Query)는 주어진 질의에 대한 시스템의 검색한 검색 결과의 수, RVD(Relevant Video data in Database)는 주어진 질의에 대해 유사성이 있어 데이타베이스로부터 반드시 검색되어져야 할 검색 결과의 수, 마지막으로 RVR (Relevant Video data that are Retrieved)는 질의에 대해 시스템이 검색한 결과 중에서 유사성을 가지고 있는 검색 결과의 수 즉, RVQ 중에서 RVD에 속하는 결과의 수라고 정의한다. 아울러, RVD를 구하기 위해 10명의 컴퓨터 관련 대학원생을 통하여 수작업를 통해 주어진 질의에 대해 비디오 데이타베이스로부터 유사성이 높은 비디오 데이타를 선택하였다. 따라서, 정확율과 재현율은 다음 식 (1)과 같다.

$$\text{정확율}(\Pr ecision) = \frac{RVR}{RVQ} \quad \text{재현율}(Recall) = \frac{RVR}{RVD} \quad \text{식(1)}$$

본 논문에서는 보다 정확한 정확율과 재현율을 측정하기 위해, 가장 널리 사용되는 11-포인트 평균 정확율과 평균 재현율을 사용한다[13]. 성능 평가는 Li 방법과 Shan의 OCMR 방법, 그리고 k-워핑 방법과 비교 수행하며, 그 중에서도 k-워핑 방법은 각도와 거리 속성을 모두 이용하기 때문에 각도에 대한 가중치(W_a)와 거리에 대한 가중치(W_d) 그리고 워핑의 수인 k를 고려해서 성능을 평가한다. 여기서, k는 적용하려는 응용 분야 즉 궤적 데이타의 특성에 영향을 받는 인자이므로 본 논문에서 사용하는 축구공 궤적 데이타의 특성상 k를 1과 2만을 이용하며, 더 많은 수는 성능 평가 결과 거의 차이가 없다. 표 4는 검색 효과 측면에서 성능을 평가한 결과를 나타낸다. 첫째, W_a와 W_d를 각각 1.0과 0.0 즉, 각도만을 고려하면서 성능 평가를 수행한 결과의 경우, 평

표 4 성능 평가 결과

		평균 정확율		평균 재현율	
	# of 워핑	k=1	k=2	k=1	k=2
W_a:W_d = 1.0:0.0	Li 방법	0.25		0.45	
	Shan 방법(OCMR)	0.30		0.44	
	k-워핑 방법	0.38	0.40	0.48	0.47
W_a:W_d = 0.7:0.3	Li 방법	0.25		0.45	
	Shan 방법(OCMR)	0.30		0.44	
	k-워핑 방법	0.44	0.45	0.46	0.47
W_a:W_d = 0.5:0.5	Li 방법	0.25		0.45	
	Shan 방법(OCMR)	0.30		0.44	
	k-워핑 방법	0.33	0.38	0.50	0.51



(a) 평균 정확율 (b) 평균 재현율

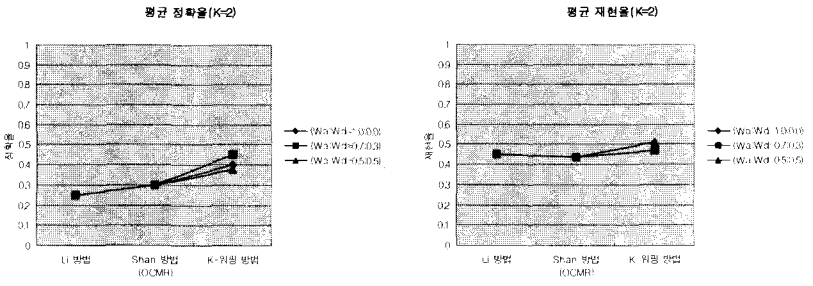
그림 11 평균 정확율 및 재현율 (k=1)

균 재현율은 거의 비슷하며 평균 정확율 측면에서는 제안하는 k-워핑 방법이 Shan의 OCMR 방법에 비해서는 약 10% 정도, Li의 방법에 비해서는 약 15% 정도의 성능 향상을 보인다. 둘째, W_a와 W_d를 각각 0.7과 0.3 즉, 각도와 거리를 모두 고려하지만 그 중에서 각도를 더 많이 고려해서 성능 평가를 수행한 결과의 경우, 평균 재현율은 거의 비슷하며 평균 정확율 측면에서는 제안하는 k-워핑 방법이 Shan의 OCMR 방법에 비해 약 15% 정도, Li의 방법에 비해 약 20% 정도의 성능 향상을 보인다. 마지막으로, W_a와 W_d를 각각 0.5와 0.5 즉, 각도와 거리를 모두 동등하게 고려해서 성능 평가를 수행한 결과의 경우, 평균 재현율은 타 방법에 비해 약간 좋으며, 평균 정확율 측면에서는 제안하는 k-워핑 방법이 Shan의 OCMR 방법에 비해서는 약 10% 정도, Li의 방법에 비해서는 약 15% 정도의 성능 향상을 보인다.

그림 11은 워핑의 수 즉 k=1일 때 각도에 대한 가중치(W_a)와 거리에 대한 가중치(W_d)에 따른 평균 정확율과 평균 재현율을 그림으로 나타낸 것이다. 평균 재현율 측면에서는 세 방법이 거의 비슷하며, 평균 정확율 측면에서 W_a와 W_d가 0.7과 0.3인 경우 즉, 각도와 거리를 모두 고려하면서도 거리보다는 각도에 더 많은 가중치를 두는 경우가 가장 좋은 결과를 보이고 있다.

그림 12는 워핑의 수 즉 k=2일때 각도에 대한 가중치(W_a)와 거리에 대한 가중치(W_d)에 따른 평균 정확율과 평균 재현율을 그림으로 나타낸 것이다. 평균 재현율 측면에서는 k=1일 때와 거의 비슷하며, 평균 정확율 측면에서는 k=1일 때보다 더 나은 결과를 보인다.

결론적으로, 제안하는 k-워핑 방법이 동등한 평균 재현율을 유지하면서 기존의 Li의 방법과 Shan의 OCMR 방법보다 평균 정확율 측면에서 더 좋은 성능을 보인다. 다시 말하면, 기존의 Li의 방법이나 Shan의 방법은 이동 객체의 궤적을 모델링하는 데 있어 각도만을 이용하는 데 반해, 제안하는 k-워핑 방법은 각도와 거리를 모두 고려하고 있으며 그 중에서 각도와 거리를 7:3으로 고려한 경우, 약 15% ~ 20% 정도의 성능 향상을 보인다. 아울러, 워핑의 수인 k는 응용 분야의 데이타 특성에 따라 다를 수 있지만, 본 논문에서 사용한 축구 비디오 데이타의 경우는 1보다는 2일 때 더 나은 성능을 보



(a) 평균 정확율 (b) 평균 재현율

그림 12 평균 정확율 및 재현율 (k=2)

인다.

5. 결론 및 향후 연구

이동 객체의 궤적은 내용 기반 비디오 검색을 위해 비디오의 내용이나 의미를 색인 하는 데 있어 매우 중요한 역할을 한다. 본 논문에서는 비디오 데이타가 지니는 이동 객체의 궤적(moving objects' trajectories)에 대한 효율적인 검색을 위해 k-워핑 알고리즘 기반 유사 부분궤적 검색(similar sub-trajectory retrieval) 기법을 제안한다. 제안하는 방법은 궤적을 구성하는 움직임 요소들 모두에 대해서 고정된 값(k)만큼까지의 반복을 허용하는 고정 반복 유사 부분궤적 검색(FRSR)과 움직임 요소들 각각에 대해서 서로 다른 값으로 할당하고 그 값만큼까지의 반복을 허용하는 가변 반복 유사 부분궤적 검색(VRSR)이다. 제안하는 방법은 이동 객체의 궤적을 모델링하기 위해 주로 사용되는 각도만의 단일 속성(property) 뿐만 아니라, 각도, 거리, 그리고 시간 간격 등을 포함하는 다중 속성(multiple properties)을 지원한다. 마지막으로, 성능 평가를 통해, 제안하는 k 워핑 알고리즘 기반 유사 부분궤적 검색 기법이 동등한 평균 재현율을 유지하면서, 기존의 Li의 방법과 Shan의 OCMR 방법보다 평균 정확율 측면에서 더 좋은 성능을 보인다. 다시 말하면, 기존의 Li의 방법이나 Shan의 방법은 이동 객체의 궤적을 모델링하는 데 있어 각도만을 이용하는 데 반해, 제안하는 k-워핑 방법은 각도와 거리를 모두 고려하고 있으며 그 중에서 각도와 거리를 7:3으로 고려한 경우가 가장 좋은 성능을 보이며, Li방법과 Shan의 방법에 비해 각각 약 15%와 20% 정도의 성능 향상을 보였다. 본 논문에서 제안하는 k-워핑 알고리즘의 장점을 정리하면 다음과 같다.

- 유사 서브시퀀스 검색을 위해서는 타임 워핑 즉, 임의의 요소에 대해 무한히 반복을 허용하는 것이 좋으나, 데이타의 특성이 다른 궤적 데이타에 대한 유사 부분궤적 검색을 위해서는 제한된 수 k번까지 반복을 허용하는 것이 더 유리하다.
- 이를 통해, 정확하지 못한 대략적인 사용자의 궤적 질의에 대해 검색의 효율성을 향상시킬 수 있다.
- 마지막으로, 질의 궤적과 데이타 궤적 사이의 유사성 측정시 일대일 매핑의 정확 매칭(exact matching)보다 더 나은 근사 매칭(approximate matching)을 효율적으로 지원할 수 있다.

앞으로의 향후 연구로는 이동 객체의 궤적 데이타가 대용량화 되어감에 따라 사용자의 질의에 대해서 빠른 검색 성능을 제공해 줄 수 있는 효율적인 색인 기법에 대한 연구을 수행하는 것이다.

참 고 문 헌

[1] A.P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao, "Modeling and Quering Moving Objects", ICDE, pp. 422-432, 1997

[2] L. Forlizzi, R.H. Guting, E. Nardelli, and M. Schneider, "A Data Model and Data Structures for Moving Objects Databases", Proc. of ACM SIGMOD Conf, pp. 319-330, 2000.

[3] R.H. Guting, M.H. Bohlen, M.Erwig, C.S. Jensen, N.A. Lorentzos, M. Schneider, and M. Vazirgiannis, "A Foundation for Representing and Querying Moving Objects", ACM Transaction on Database Systems, Vol. 25, No. 1, pp. 1-42, 2000.

[4] J.R. Smith and S.F. Chang, "VisualSEEk: a Fully Automated Content-Based Image Query System," in Proceedings of ACM Multimedia 96, pp. 87-98, 1996.

[5] Virginia, E.Ogle, and M. Stonebraker, "Chabot: Retrieval from a Relational Database of images," IEEE Computer, Vol. 28, No. 9, pp. 40-48, 1995.

[6] J.Z. Li, M.T. Ozsu, and D. Szafron, "Modeling Video Temporal Relationships in an Object Database Management System," in Proceedings of Multimedia Computing and Networking(MMCN97), pp. 80-91, 1997.
[7] J.Z. Li, M.T. Ozsu, and D. Szafron, "Modeling of Video Spatial Relationships in an Objectbase *Management System," in Proceedings of* International Workshop on Multimedia DBMS, pp. 124-133, 1996.
[8] M.K. Shan and S.Y. Lee, "Content-based Video Retrieval via Motion Trajectories," in Proceedings of SPIE Electronic Imaging and Multimedia System II, Vol. 3561, pp. 52-61, 1998.
[9] B.K. Yi, H.V. lagadish, and C. Faloutsos, "Efficient Retrieval of Similar Time Sequences Under Time Warping," In Proc. Int'l. Conf. on Data Engineering, IEEE, pp. 201-208, 1998.
[10] S.H. Park *et al.,"Efficient Searches for Simialr* Subsequence of Difference Lengths in Sequence Databases," In Proc. Int'l. Conf. on Data Engineering. IEEE, pp. 23-32, 2000.
[11] S.W. Kim, S.H. Park, and W.W. Chu, "An Index-Based Approach for Similarity Search Supporting Time Warping in Large Sequence Databases," In Proc. Int'l. Conf. on Data Engineering. IEEE, pp. 607-614, 2001.
[12] G. Salton, "A New Comparison between Conventional Indexing(MEDLARS) and Automatic *Text Processing(SMART)," Journal of the* American Society for Information Science, Vol. 23, No. 2, pp 75-84, 1972.
[13] G. Salton and M. McGill, *An introduction to Modern Information Retrieval,* McGraw-Hill, 1993.

심 춘 보
1996년 전북대학교 컴퓨터공학과(공학사). 1998년 전북대학교 컴퓨터공학과(공학석사). 2003년 전북대학교 컴퓨터공학과(공학박사). 관심분야는 멀티미디어 데이터베이스, 멀티미디어 정보검색, 데이타 접근(색인) 기법, 시공간 데이타베이스

장 재 우
1984년 서울대학교 전자계산기공학과(공학사). 1986년 한국과학기술원 전산학과(공학석사). 1991년 한국과학기술원 전산학과(공학박사). 1996년~1997년 Univ. of Minnesota, Visiting Scholar. 1991년 ~ 현재 전북대학교 컴퓨터공학과 교수. 관심분야는 멀티미디어 데이타베이스, 멀티미디어 정보검색, 하부저장구조, 시공간 데이타베이스