비디오 데이터에서 움직임 경로의 효율적인 검색을 위한 시그니처-기반 접근 기법

심춘보[0] 장재우
전 북 대 학 교 컴 퓨 터 공 학 과
{cbsim, jwchang}@dblab.chonbuk.ac.kr

A Signature-based Access Method for Efficient Retrieval on Moving Objects' Trajectories in Video Data

Choon-Bo Shim[0] Jae-Woo Chang

Dept. of Computer Engineering, Chonbuk National University

요 약

본 논문에서는 비디오 데이터가 지니는 움직임 객체의 움직임 경로(Moving Objects' Trajectories)를 이용한 사용자 질의에 대해 효율적인 검색을 위한 새로운 시그니처-기반 접근 기법을 제안한다. 제안하는 시그니처 기반 접근 기법은 데이터 파일을 직접 접근하기 전에 전체 시그니처들을 탐색하여 필터링을 수행하기 때문에, 순차 탐색에 비해 디스크 접근 횟수를 감소시켜 검색 성능을 향상시킨다. 마지막으로, 성능 평가를 통해 제안하는 방법이 삽입 시간, 검색 시간, 그리고 부가 저장 공간의 검색 효율(Retrieval Efficiency) 측면에서 성능이 우수함을 보인다.

1. 서론

대용량의 멀티미디어 데이터베이스에서 사용자의 다양한 요구를 보다 효과적으로 처리하기 위해서는 효율적인 내용 기반 멀티미디어 검색이 요구된다. 멀티미디어 데이터 중에서 텍스트나 이미지 데이터와는 달리 비디오 데이터가 지니는 중요한 특징은 움직임 객체에 대한 움직임 정보이다. 이러한 움직임 정보는 각각의 프레임 내에서의 객체들간의 공간적인 정보와 일련의 프레임들간의 시간적인 정보가 결합된 시공간 관계성을 통해 표현될 수 있으며, 비디오 데이터에 대한 사용자의 내용 및 개념 기반 검색을 수행하는 데 있어 매우 중요한 역할을 한다. 한편, 내용 기반 비디오 검색에 관한 연구는 비디오 데이터로부터 추출된 의미 및 내용 정보를 효율적으로 모델링 할 수 있는 데이터 표현 기법[1,2,3]에 관심이 집중되었다. 그러나, 대용량의 멀티미디어 데이터를 다루기 위해서는 연구의 주된 관심이 사용자의 다양한 질의에 대한 빠른 검색 성능을 제공할 수 있는 검색의 효율성에 있다.

따라서, 본 논문에서는 움직임 경로를 이용한 다양한 사용자 질의에 대해 효율적인 검색을 위한 시그니처-기반 접근 기법(Signature-based Access method for Moving Objects' Trajectories, SAMOT)을 제안한다. 제안하는 시그니처-기반 접근 기법은 데이터 파일을 직접 접근하기 전에 전체 시그니처[4]들을 탐색하여 필터링을 수행하기 때문에 순차 탐색에 비해 디스크 접근 횟수를 감소시켜 검색 성능을 향상시킨다.

본 논문의 구성은 다음과 같다. 2장에서는 검색의 효율성을 위해 제안하는 시그니처-기반 접근 기법에 대해서 기술하고, 3장에서는 제안하는 방법의 우수성을 입증하기 위해 검색 효율측면에서 성능 비교를 수행한다. 마지막으로, 4장에서 결론 및 향후 연구 방향을 제시한다.

2. 시그니처-기반 접근 기법(SAMOT)

2.1 SAMOT의 전체적인 구조

SAMOT 접근 기법은 크게 두 부분으로 나뉜다. 즉, 비디오 샷으로부터 추출된 움직임 객체의 움직임 정보들을 저장하고 있는 데이터 파일 부분과 움직임 정보에 대한 시그니처를 생성하여 저장하는 시그니처 파일 부분이다. 먼저 데이터 파일의 자료 저장 구조는 그림 1과 같다.

```
struct _mot {
   int id;                          // 식별자
   int num;                         // 움직임의 수
   int locList[MAX_LOC];            // 움직움이 발생한 위치 정보
   char objList[MAX_OBJ][MAXNAME]   // 위치 정보에서의
                                    객체 정보
   Motion motionList[MAX_MOTION];   // 움직임 Motion 정보
   RecID v_id;          // 실제 비디오 샷을 가리키는 식별자
} T_MOT;
```

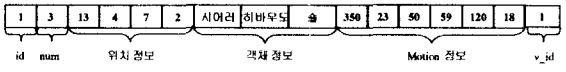


그림 1 데이터 파일의 자료 저장 구조

여기서, id는 비디오 샷으로부터 추출된 움직임 정보를 저장하는 데이터 파일에서의 레코드 식별자를 의미하고, num은 움직임 정보를 이루는 Motion의 갯수를 나타낸다. 위치정보는 움직임 객체의 위치 정보 즉, 그림3에서와 같이 본 논문에서 제안하는 축구 비디오 데이터를 위한 운동장 모델에서의 위치 정보를 의미한다. 객체 정보는 각각의 위치 정보에서 움직임 객체를 소유하고 있는 행위자(Actor) 즉, 축구 비디오 데이터베이스의 경우 축구공을 가지고 있는 선수를 의미한다. Motion 정보는 움직임 객체가 움직인 방향(Real Angle)과 거리(Distance)의 쌍으로 구성된다. 마지막으로, v_id는 해당 움직임 정보가 추출된 실제 축구 비디오 데이터 스트림을 가리키는 객체 식별자를 나타낸다.

움직임 정보를 위한 시그니처 파일 부분은 세부적으로 4개의 파일로 구성된다. 첫째, 해당 움직임 객체의 움직임 정보들 중에서 객체 정보에 해당하는 시그니처를 생성해서 저장하는 객체 시그니처 파일 둘째, 위치 정보에 해당하는 시그니처를 생성해서 저장하는 위치 시그니처 파일 셋째, 위치 정보와 순서 정보(Order Information)를 포함해서 시그니처를 생성해서 저장하는 순서 시그니처 파일 마지막으로, 시그니처 파일과 데이터 파일을 연결하기 위한 링크 파일이다. 본 논문에서 제안하는 SAMOT 접근 기법의 전체적인 구조는 그림 2와 같다.

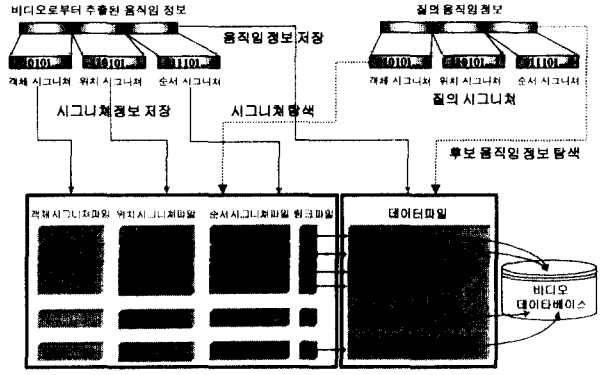


그림 2 SAMOT 접근 기법의 전체적인 구조

2.2 시그니쳐 생성

본 논문에서는 움직임 객체의 움직임 정보에 대해서 각각 객체 시그니쳐, 위치 시그니쳐, 그리고 순서 시그니쳐로 나누어 시그니쳐를 생성한 후, 각각의 시그니쳐 파일에 분리해서 저장한다. 이렇게 함으로써 사용자의 다양한 질의를 분석해서 질의에 부합하는 해당 시그니쳐 파일만을 탐색함으로써 전체적인 검색 성능을 떨어뜨리지 않도록 하기 위함이다. 시그니쳐를 생성하는 방법은 다음과 같다.

첫째, 움직임 정보들 중에서 객체 정보에 기반해서 생성한 객체 시그니쳐는 움직임 정보를 이용한 사용자의 질의 중에서 객체 정보를 이용한 질의를 효율적으로 처리하기 위함이며, 객체 정보를 입력으로 해쉬함수를 이용해서 각각의 객체들에 대한 시그니쳐를 생성한다.

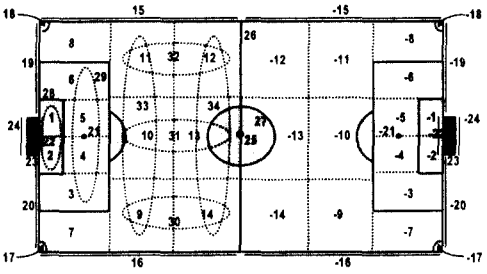


그림 3 축구 비디오 데이터를 위한 축구 경기장 모델

둘째, 위치 시그니쳐를 생성하는 방법은 그림 3에서와 같이 축구 경기장 모델을 기반으로 축구공의 움직임이 발생한 위치에 해당하는 번호에 따라 해당 비트를 1로 세팅시키는 방법이다. 그림 4는 위치 시그니쳐를 생성하는 예를 보여준다.

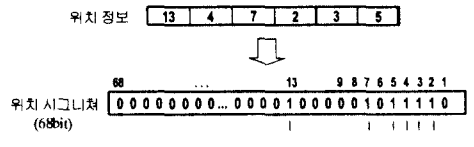


그림 4 위치 시그니쳐 생성 예

마지막으로, 순서 시그니쳐는 움직임 객체의 움직임 정보들 중에서 움직임이 발생한 위치와 더불어 순서를 같이 고려한다. 순서 시그니쳐를 생성하는 방법은 먼저, 그림 3에서와 같이 축구 운동장 모델에서의 위치 정보 68가지에 대해서 각각 하나의 아스키 문자로 매핑시킨 후, 순서 정보를 결합해서 이를 문자열로 변환하여 순서 시그니쳐를 생성한다. 예를 들어 축구 비디오 샷으로부터 추출된 축구공의 움직임 위치가 다음과 같이 "4 10 2 5 3"으로 이루어져 있다면, 각각 위치에 맞는 해당 아스키 문자를 다음과 같이 'D', 'J', 'B', 'E', 'C' 생성한다. 그런 후에, 여기에 순서 정보를 같이 결합해서 다음과 같이 "D1", "J2", "B3", "E4", "C5"과 같은 최종적인 순서 정보 리스트를 생성하고, 이를 토대로 순서 시그니쳐를 생성한다.

2.3 삽입 및 검색 과정

주어진 움직임 객체의 움직임 정보들을 그림 2의 SAMOT 접근 기법에 삽입하기 위한 과정은 다음과 같다. 먼저, 움직임 정보들을 기반으로 앞에서 언급한 시그니쳐 생성 알고리즘을 이용해 객체 시그니쳐, 위치 시그니쳐, 그리고 순서 시그니쳐를 생성한 후, 생성된 시그니쳐들을 각각의 해당 시그니쳐 파일에 삽입한다. 둘째, 실제 움직임 객체의 움직임 정보를 데이터 파일에 삽입한다. 마지막으로, 실제 데이터 파일에 있는 해당 움직임 정보들을 접근하기 위해 필요한 링크 파일에 데이터 파일에서의 해당 움직임 정보의 저장 위치와 크기 정보를 저장하는 것으로 삽입 과정을 마친다.

한편, 움직임 객체의 움직임 정보를 이용한 사용자의 질의 처리는 다음과 같다. 먼저, 질의 움직임 정보가 주어지면, 질의 움직임 정보를 이용해서 질의 시그니쳐 즉, 질의 객체 시그니쳐, 질의 위치 시그니쳐, 그리고 질의 순서 시그니쳐를 생성한다. 이렇게 생성된 질의 시그니쳐들을 이용하여 시그니쳐 파일에 저장된 각각의 시그니쳐들을 순차적으로 탐색하여 필터링을 수행한다. 시그니쳐 파일을 탐색하는 경우 사용자 질의 타입에 따라 질의를 처리하기 위해 필요한 시그니쳐 파일만을 탐색한다. 물론, 모든 시그니쳐 파일을 모두 탐색해야 할 경우는 먼저 객체 시그니쳐 파일로부터 후보 시그니쳐들만을 선택한다. 그리고 선택된 후보들만을 이용해서 두 번째로 위치 시그니쳐 파일을 탐색한다. 그리고 마지막으로는 두 단계의 시그니쳐 파일 탐색을 거쳐서 얻은 후보들을 이용해서 순서 시그니쳐 파일을 탐색한다. 이렇게 시그니쳐 파일 탐색을 통해 최종적으로 선택된 후보 시그니쳐만을 가지고 데이터 파일을 접근해서 해당하는 실제 움직임 정보들을 탐색한다. 시그니쳐 기법의 특성상 검색 매치 오류(False Drop)[4]를 감안해서 검색 매치 오류를 체크한 후, 최종적으로 주어진 사용자 질의를 만족하는 최종 결과를 검색한다.

3. 실험 및 성능평가

본 논문에서는 128MB 메인 메모리를 탑재한 펜티엄 III-800의 Windows 2000 운영체제 하에서 실험한다. 아울러, 성능 평가를 위한 실험 데이터 셋은 균등 분포 특성을 가진 임의 데이터 100,000건으로 구성된 Synthetic 데이터 셋과 "골인되는(Goal In) 장면"을 포함하고 있는 실제 축구 비디오 데이터 350개를 확장시켜 생성한 460,000건으로 구성된 Real 데이터 셋이다. 이 두 가지의 데이터 셋은 모두 1개에서 15개의 움직임 정보들(Motion)로 구성되어 있다. 성능 평가는 삽입 시간(Insert Time), 검색 시간(Retrieval Time), 그리고 부가 저장 공간(Storage Overhead)으로 나누어서 수행한다. 그리고 움직임 객체의 움직임 경로를 위한 기존 연구에서는[1,2,3] 빠른 검색 성능을 지닌 접근 기법을 사용하지 않기 때문에 본 논문에서는 순차 접근 방식(Sequential Scan Method : SeqScan)과 비교해서 성능을 평가한다.

먼저, 삽입 시간은 실제 축구 비디오 샷으로부터 추출된 움직임 정보들로 구성된 데이터를 시그니쳐 파일과 데이터 파일에 각각 저장하는 데 소요되는 시간을 의미한다. 그림 5는 각각 Synthetic 데이터 셋과 Real 데이터 셋을 삽입하는 데 소요되는 시간을 나타낸다. 여기서, 세로축은 삽입 시간을 초

단위로 나타낸 것이며, 삽입 시간은 메모리에서의 연산 시간(CPU 시간)과 디스크 접근 시간을 모두 포함한 전체 시간(Wall 시간)을 의미한다. Real 데이터 셋의 경우 SeqScan 방식이 삽입 시간은 약 27초, SAMOT 접근 기법은 약 87초 정도 소요된다. SAMOT 접근 기법이 삽입 시간이 느린 이유는 움직임 정보들을 데이터 파일에 저장하기 전에 시그니처를 구성한 후, 구성된 시그니처들을 각각의 시그니처 파일에 저장하는 시간을 요구하기 때문이다.

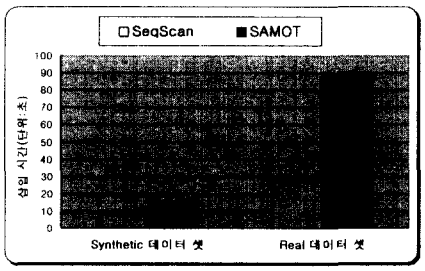
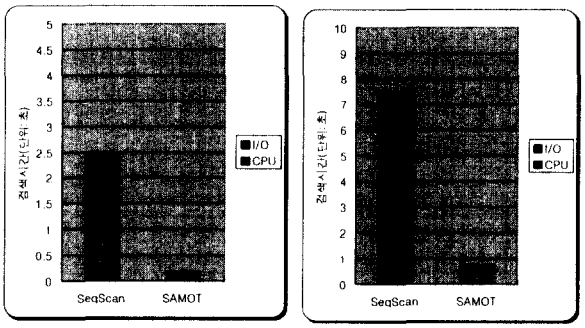


그림 5 삽입 시간

둘째, 검색 성능을 평가하기 위해 Synthetic 데이터 셋과 Real 데이터 셋으로부터 무작위로 각각 100개의 질의 움직임 정보를 추출한 것을 이용해 평균값을 구해서 검색에 요구되는 검색 성능을 측정한다. 그림 6은 검색 질의에 대한 전체 응답 시간을 메모리에서 계산하는 데 소요되는 CPU 시간과 데이터 파일에서 움직임 정보를 읽어오는 데 걸리는 디스크 접근 I/O 시간으로 나누어 검색 시간을 비교 평가한 것이다. 그림 6의 (a)는 Synthetic 데이터 셋에 대한 검색 시간을 측정한 결과로 SeqScan 방식의 경우 CPU 시간과 I/O 시간 모두 약 1.2초 걸리는 데 반해 SAMOT 접근 기법의 경우 CPU 시간은 약 0.2초, I/O 시간은 0.05초 정도 밖에 소요되지 않는다. 그림 6의 (b)는 Real 데이터 셋에 대한 검색 시간으로 Synthetic 데이터 셋과 거의 비슷한 결과를 보인다. 전체 시간을 고려해 볼 경우 SAMOT 접근 기법이 SeqScan 방식에 비해 약 10배 정도의 검색 성능을 보인다.



(a) Synthetic 데이터 셋 (b) Real 데이터 셋

그림 6 검색 시간(CPU시간 + I/O시간)

부가 저장 공간 비율(Storage Overhead : SO)은 식 (1)과 같이 원래의 데이터 파일의 크기를 기준으로 색인 파일에서 부가적으로 더 요구되는 저장공간의 비율을 의미한다. 여기서 색인 파일의 크기는 객체 시그니처 파일, 움직임 시그니처 파일, 순서 시그니처 파일 그리고 링크 파일과 모두 합한 크기를 의미한다.

$$SO = \frac{\text{색인파일의크기}}{\text{데이타파일의크기}} * 100 \quad \text{... 식 (1)}$$

SeqScan 방식의 부가 저장 공간 비율이 0%인데 비해, SAMOT 접근 기법은 Synthetic 데이터 셋의 경우 약 16%의 부가 저장 공간을 요구하고, Real 데이터 셋의 경우는 약 27%정도의 부가 저장 공간을 요구한다. 시그니처 파일의 특성상 타 접근 기법에 비해 적은 부가 저장 공간을 요구한다.

표 1은 SAMOT 접근 기법을 구성하는 시그니처에 따른 필터링 효과를 나타낸다. 위치 시그니처를 통해 필터링된 후 얻은 후보 결과는 Synthetic 데이터 셋의 경우 100,000건의 시그니처 중에서 평균 2,181건만이 후보 결과로 남는다. 그에 반해, 위치 시그니처와 순서 시그니처에 의해 필터링되는 경우는 평균 638건만이 후보 결과로 남는다. 마지막으로 638건의 후보 시그니처들을 이용해서 데이터 파일을 탐색하여 질의 움직임 정보와 관련이 없는 검색 오류 매치 11건을 제외한 나머지 평균 627건이 최종적으로 남은 결과이다.

표 1 SAMOT을 통해 검색된 후보 결과

시그니처 / 데이터 셋	위치 시그니처	위치 + 순서 시그니처	최종 결과
Synthetic 데이터 셋	2181	638	627
Real 데이터 셋	10358	800	796

4. 결론

본 논문에서는 검색의 효율성을 최대화할 수 있는 움직임 객체를 위한 시그니처-기반 접근 기법을 제안한다. 제안하는 SAMOT 접근 기법은 데이터 파일을 직접 접근하기 전에 전체 시그니처들을 탐색하여 필터링을 수행하기 때문에 순차 탐색에 비해 많은 수의 디스크 접근 횟수를 감소시킴으로써 검색 성능을 향상시킨다. 아울러, 제안한 방법의 효율성을 측정하기 위해, 검색 효율 측면을 고려해서 성능 평가를 수행하였다. 삽입 시간의 경우, 제안하는 SAMOT 접근 기법이 시그니처를 구성한 후, 구성된 시그니처 정보를 시그니처 파일에 저장하는 시간을 요구하기 때문에 SeqScan에 비해 더 많은 시간을 요구한다. 그에 반해, 검색 시간의 경우, 제안하는 SAMOT 접근 기법이 SeqScan에 비해 Synthetic 데이터의 경우 약 10배 정도, Real 데이터의 경우도 약 10배 정도의 성능 향상을 보인다.

향후 연구 과제는 본 시스템을 통해 사용자가 다양한 질의를 할 수 있고 또한, 편리하게 검색 결과를 브라우징 할 수 있는 내용 및 개념 기반 축구 비디오 검색 GUI(Graphic User Interface)를 구현하는 것이다.

참고문헌

[1] John Z. Li, M. Tamer Ozsu, Duane Szafron, "Modeling Video Temporal Relationships in an Object Database Management System," in Proceedings of Multimedia Computing and Networking(MMCN97), pp. 80-91, 1997.

[2] Man-Kwan Shan and Suh-Yin Lee, "Content-based Video Retrieval via Motion Trajectories", In Proceedings of the International Conference on SPIE, Vol. 3561, pp. 52-61, 1998

[3] Z.Aghbari, K.Kaneko, A.Makinouchi, "Modeling and Querying Videos by Content Trajectories", In Proceedings of the International Conference and Multimedia Expo, pp. 463-466, 2000

[4] C. Faloutsos and S. Christodoulakis, "Signature files : An access methods for documents and its anylytical performance evaluation," ACM Transaction on Database Systems, Vol. 2, No. 4 pp. 267-288, 1984.