측위를 위한 결정 트리 구현

임재걸 정승환°
동국대학교 컴퓨터멀티미디어학과
{yim, honourj°}@dongguk.ac.kr

Implementation of a Decision Tree for Positioning

Jaegeol Yim, Seunghwan Jeong°
Dept of Computer and Multimedia, Dongguk University

요 약

무선 통신 기술의 발달로 사용자의 이동성이 제공되기 시작하면서 위치기반서비스(LBS: Location Based Service)가 부각되었다. 서비스의 예로 공공안전 서비스, 위치추적 서비스, 항법 서비스, 정보제공 서비스 등 부가가치가 높은 서비스들이 많이 있는데, 이러한 서비스를 개발 하려면 필수적으로 사용자의 위치를 파악해야 한다. 옥내 측위 방법으로 여러 가지가 실험되고 있는데, Fingerprint 방법이 일반적으로 가장 정확도가 높다. 기존의 Fingerprint 방식에는 K-NN 방법과 Bayesian 방법이 소개되었는데, 결정 트리를 이용한 방법은 효율성이 기존의 K-NN이나 Bayesian 방법보다 뛰어나게 좋음에도 불구하고 적용한 사례가 없다. 그래서 본 논문은 결정 트리를 이용하는 방법을 제안한다. K-NN 및 Bayesian 방법과 제안하는 방법을 비교 분석한 결과와 제안하는 방법의 실험 결과도 보인다.

1. 서론

무선 통신 기술의 발달로 사용자의 이동성이 제공되기 시작하면서 위치기반서비스(LBS: Location Based Service)가 부각되었다. 위치기반서비스는 이동통신망이나 위성항법장치(GPS) 등을 통해 얻은 위치정보를 기반으로 사용자에게 유용한 서비스를 제공하는 것을 말한다. 서비스의 예로 공공안전 서비스, 위치추적 서비스, 항법 서비스, 정보제공 서비스 등 부가가치가 높은 서비스들이 많이 있는데, 이러한 서비스를 개발 하려면 필수적으로 사용자의 위치를 파악해야 한다.

옥내용 측위 시스템은 여러 가지 방법으로 위치추정 연구가 진행되고 있는데, 기존의 옥내 측위 방법은 RF(radio frequency) 신호와 초음파 신호의 도착 시간의 차이를 이용한 방법 (CRICKET[1]), 적외선이 벽을 통과하지 못한다는 사실을 이용하는 (Active Bat[2]) 그리고 후보지점의 특징 값들을 이용하는 Fingerprint 방법(RADAR[3]), 등이 있다.

Fingerprint 방식은 준비 단계와 실시간 단계로 구성된다. 준비 단계에서는 각 위치에서 특징 데이터를 미리 측정하여 Training Data를 생성하며, 실제 실시간 단계에서는 그 장소에서 측정한 특징 데이터를 가지고 Training Data를 참조하여 현재 위치를 추정한다. 기존의 RADAR에서는 특징 데이터로 UDP 패킷의 신호의 세기를 사용하였다. 이를 위하여 RADAR에서는 base station을 특별히 두었는데, 본 논문에서는 WLAN (무선랜)을 위하여 이미 설치된 AP의 신호의 세기를 특징 데이터로 삼는다.

기존의 Fingerprint 방식에는 K-NN[4]과 Bayesian[5] 방법이 있는데, 결정 트리를 이용하는 방법은 효율성이 기존의 K-NN이나 Bayesian 방법보다 뛰어나게 좋으면서도 적용한 사례가 없었다. 그래서 본 논문은 결정 트리를 이용하는 방법을 제안하며, K-NN, Bayesian 방법과 효율성을 비교 분석한다. 제안하는 방법을 구현하여 실험적으로 제안하는 방법의 정확도를 검증한다.

2. 기존 연구

기존의 K-NN 방법과 Bayesian 방법을 알아보며, 본 논문에서 제안하는 결정 트리 방법을 설명한다.

2.1 K-Nearest Neighbor 방법

실시간 단계에서 측정한 특징 데이터를 가지고 준비 단계에서 작성한 Training Data 중 가장 가까운 이웃지점 k개의 평균으로 현재 위치를 판정하는 방법으로 RF(radio frequency) 신호의 세기를 특징 데이터로 이용한 RADAR를 예로 들 수 있다.

2.2 Bayesian classification 방법

특징 데이터의 일반 형식을 $X=(x_1, x_2, \ldots, x_n)$라 하자. 본 논문에서는 x_i는 i 번 AP의 RSSI 측정값이다. 측위에서는 Training Data를 획득한 지점, 즉 후보지점이 Class가 된다. 후보지점을 각각 $C_1, C_2 \ldots, C_m$이라 하자. 실시간 단계에서 측정한 샘플 X 가 Class C_i 에 속할 확률을 구하는 식은 $P(C_i|X)=\frac{P(X|C_i)P(C_i)}{P(X)}$이다. 이 식에서 $P(X)$는 상수이고, 모든 후보지점에 대하여 사용자가 현재 위치할 확률이 동등, 즉 $P(C_1) = P(C_2) = \ldots = P(C_m)$ 이므로, $P(X|C_i)$가 최대인 C_i가 사용자의 현재 지점이 된다. $P(X|C_i)$는 아래 식에 의해서 구해진다.

$$P(X|C_i)=\prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i)\times P(x_2|C_i)\times \ldots \times P(x_n|C_i)$$

만약 A_k가 이산적인 값을 가지면, $P(x_k|C_i)=\frac{s_{ik}}{s_i}$이며, 이때 s_{ik}는 C_i Class의 Training Sample들 중 A_k의 값이 x_k 인 샘플들의 수이고, s_i는 C_i에 속하는 Training Sample들의 수이다.

2.3 결정 트리 방법

<표 1> Training Data를 이산화한 Table 예

AP1	AP2	AP3	AP4	AP5	Class
interval1	interval2	interval2	interval3	interval1	지점1
interval1	interval1	interval2	interval2	interval3	
interval2	interval3	interval1	interval3	interval1	
interval3	interval3	interval2	interval2	interval2	지점2
interval2	interval3	interval1	interval2	interval2	
interval1	interval2	interval3	interval1	interval1	
interval1	interval1	interval1	interval3	interval1	지점3
interval2	interval2	interval2	interval2	interval3	
interval3	interval1	interval1	interval3	interval3	

<표 1>과 같은 이산화된 Table이 있다고 가정하고 이 테이블을 Root로 하여 결정 트리를 생성하는 과정은 아래와 같다.

다섯 개의 AP 중, Gain이 가장 큰 AP_i를 현재 노드의 레이블로 하고, AP_i의 도메인이 $interval_1, \ldots, interval_l$라면, 이에 속하는 각 $interval_h$에 대하여 $interval_h$를 레이블로 하는 가지를 만들고 이 가지에 AP_i의 값이 $interval_h$인 모든 행으로 구성된 테이블에서 AP_i 열을 삭제하여 작성한 테이블을 자식노드로 생성한다.

<표 1>에서 AP_i 요소에 대한 Gain은 [식 1]처럼 테이블의 Information(정보이득)에서 AP_i의 Entropy를 뺀 값이다.

$$Gain(AP_i) = I(S_1, S_2, \ldots, S_m) - E(AP_i) \cdots\cdots\cdots [식\ 1]$$

여기서 $I(S_1, S_2, \ldots, S_m)$는 테이블을 분류하는데 필요한 정보의 양으로써, 1부터 m까지의 Class에 대하여 S_i는 i번째 Class에 속하는 샘플의 개수를 뜻한다. P_i는 i번째 Class가 나올 확률이라고 하면 Information은 [식 2]처럼 구한다.

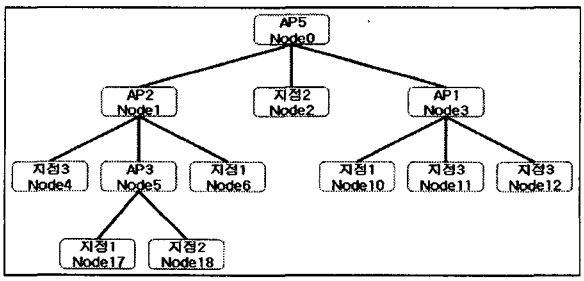
$$I(S_1, S_2, \ldots, S_m) = -\sum_{i=1}^{m} P_i \log_2(P_i) \cdots\cdots\cdots\cdots [식\ 2]$$

그리고 $E(AP_i)$는 현재 테이블을 AP_i를 기준으로 분류하였을 때, 분류 결과를 분류하는데 필요한 정보의 양으로 [식 3]으로 구한다.

$$E(AP_i) = \sum_{j=1}^{v} \frac{S_{1j} + \ldots + S_{mj}}{S} I(S_{1j}, \ldots, S_{mj}) \cdots\cdots\cdots [식\ 3]$$

생성된 각각의 자식 테이블들은 다시 Root가 되어 위에 설명한 방법과 동일하게 자식 테이블을 생성한다.

더 이상 자식 테이블을 생성할 수 없거나, 자식 테이블을 구성하는 모든 샘플이 동일한 Class에 속하면 현재 테이블은 결정 트리의 단말 노드에 해당한다. (그림 1)은 <표 1>에 대한 결정 트리의 결과이다.



(그림 1) <표 1>에 대한 결정 트리

3. 측위 프로그램 개발

본 절에서는 결정 트리를 이용한 측위 프로그램의 구현 환경과 프로그램의 구성에 대하여 설명한다.

3.1 개발 환경

결정 트리를 이용한 측위 프로그램을 노트북 컴퓨터에 구현한다. 측위 프로그램은 준비단계에서 결정 트리를 작성하고 실시간 단계에서 랜 카드가 AP(Access Point)로부터 수신되는 신호의 강도를 읽은 후, 이를 바탕으로 현재 위치를 판정한다. 사용하는 랜 카드는 Intel(R) PRO/Wireless 2200BG Network Connection이고, 프로그램 개발 도구로는 Microsoft Visual C# 2005를 사용한다.

3.2 결정 트리를 이용한 측위 프로그램

<표 2>는 이산화된 Table을 가지고 결정 트리를 생성하는 알고리즘을 보이고, <표 3>은 실시간 단계에서 측정치로 결정 트리를 참조하여 현재위치를 판정하는 알고리즘을 보인다.

<표 2> 결정 트리를 생성하는 알고리즘

```
algorithm CreatDesisionTree(int[][] Table,
                            ListType MacList, int Index)
재귀 호출을 통해 배열을 이용한 결정 트리를 생성한다.
사전조건 : Table은 Root 테이블 또는 부모 테이블이다.
           MacList는 Table의 AP에 대한 MacAddress.
           Index는 Table이 위치한 실제 Tree의 노드번호.
           Tree배열은 적역변수이다.
사후조건 : subTable, subMacList, subIndex를 생성하여
           재귀호출 한다.
           subIndex는 subTale이 위치한 실제 Tree의 노드번호.
           Tree의 단말노드에 도달하기 까지 중간 노드를 구한다.
1. countOfIdentity = 0
2. if(Table 행의 개수 != 1)
   1. if (Table을 구성하는 모든 행의 class 값, 즉 '지점'
         이 동일할 경우)
      1. Tree[Index]에 '지점' 저장
      2. return
   2. else if(Table이 행의 개수 == 0)
      1. Tree[Index] = Empty
      2. return
   3. else if(Table의 열의 개수 == 1)
      1. Tree[Index]에 '지점'을 비율로 저장
      2. return
   4. end if
3. else
   1. Tree[Index]에 class 값, 즉 '지점' 저장
   2. return
4. end if
5. Table에 대한 Information을 구한다.
6. Table의 AP마다 Entropy를 구한다.
7. Tree[Index]에 Gain이 가장 큰 AP의 MacAddress와 배열
   P 저장. 배열 P는 Table에 출현하는 각 지점의 확률로 구성됨.
8. subMacList 생성
9. loop(i=1; i<=Interval 개수; i++)
   1. subTable 생성
   2. subIndex = Index*Interval 개수 + i
   3. CreatDesisionTree(subTable, subMacList, subIndex)
10. end loop
end CreatDesisionTree
```

4. 실험

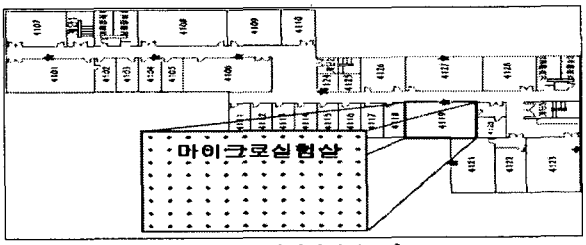
본 절에서는 제안한 방법의 실험 결과를 바탕으로 제안하는 방법의 정확도를 보이고, 기존의 방법과 효율성을 비교 분석한다.

<표 3> 현재위치를 판정하는 알고리즘

```
algorithm Locate(MacType items[])
결정 트리를 이용하여 현재위치를 판정한다.
사전조건 : Tree[]는 배열을 이용한 결정 트리로 전역변수이다.
           items[]에는 실시간 단계에서 측정된 MacAddress와
           신호세기가 저장되어 있다.
사후조건 : 단말 노드의 지점, 즉 현재 위치를 반환한다.
1. indexOfNode = 0
2. loop(Tree의 레벨 수만큼)
   1. if(BoolMacAddress(Tree[indexOfNode]))
      // Tree[indexOfNode]가 MacAddress이면 참임.
      1. Interval = funcInt (items[], Tree[indexOfNode);
      // items[]에서 MacAddress의 RSSI를 찾아, RSSI의
         구간을 구하여 줌.
      2. IndexOfNode = Interval에 해당하는 자식노드의 Index.
   2. else if (Tree[indexOfNode]가 Empty 이다.)
      1. indexOfNode = 부모노드의 Index
      2. return(Tree[indexOfNode]의 확률 배열 P)
   3. else
      1. return(Tree[indexOfNode] ) // return 지점
   4. end if
3. end loop
4. Return Tree[indexOfNode];
end Locate
```

4.1 실험 환경

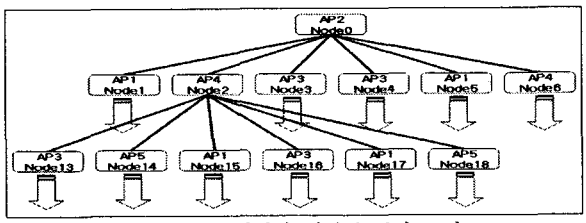
제안하는 측위 프로그램의 정확도를 보이기 위하여 (그림 2)에 보이는 자연과학관 4층에서 실험을 실시한다. (그림 2)에는 총 8개의 AP 설치 지점이 '★'로 표시되었다. (그림 2)의 마이크로 실험실처럼 자연과학관 4층에 1M 간격마다 후보지점을 설정하여, 준비단계에서는 각각의 지점에 대하여 30회씩 샘플을 측정함으로써 Training Data를 작성한다.



(그림 2) 자연과학관 4층

4.2 실험 결과

(그림 2)에 보이는 자연과학관 4층에서 준비단계 때 각 위치마다 미리 측정한 Training Data를 6개 interval로 이산화하여 결정 트리를 생성하면 (그림 3)과 같은 결정 트리를 얻는다. 실제 실시간 단계에서 측정한 데이터를 가지고 (그림 3)의 결정 트리를 이용하여 위치를 추정하는 실험을 100회 수행한 결과 평균 오차거리는 3.241m 이었다.



(그림 3) 설험결과 생성된 결정 트리

4.3 복잡도 분석

K-NN 방법, Bayesian 방법 결정 트리 방법에 대하여 실제 실시간 단계에서 위치 추정을 하는데 소요되는 수행시간은 Training Data에 대한 Interval의 수, AP의 수, 후보지점의 수에 대한 함수이다.

K-NN 방법의 경우 실시간 단계에서 측정한 각 AP의 신호세기를 Training Data의 각각의 후보지점에 대해서 작성된 각 AP의 신호세기 평균과 비교하여 차이가 가장 작은 후보지점을 실측 위치라고 판정하기 때문에 수행 시간은 Training Data의 'AP의 수 * 후보지점의 수' 만큼 걸린다. Bayesian 방법의 경우 K-NN 방법과 비슷한데, Bayesian 방법의 Training Data는 각각의 후보지점에 대해서 측정된 각 AP의 신호세기가 해당하는 Interval 별로 카우트 되어있다. 그래서 실시간 단계 때 측정한 각 AP의 신호세기가 Training Data의 각각의 후보지점에 대해서 각 AP마다 해당하는 Interval과 Interval의 카운트된 수치를 찾아야 함으로 수행 시간은 Training Data의 'Interval의 수 * AP의 수 * 후보지점의 수' 만큼 걸린다. 결정 트리 방법의 경우에는 결정 트리의 root에 해당하는 AP의 실시간 단계에서 측정한 측정치가 속한 Interval로 자식노드를 찾은 다음, 이 자식노드를 root로 하여, 위의 과정을 단말 노드에 도착할 때까지 반복하기 때문에 결정 트리 방법은 'Interval의 수 * AP의 수' 만큼의 수행시간이 걸린다.

5. 결론

Interval의 수를 i, AP의 수를 m, 후보지점의 수를 n이라 할 때, (본 실험의 경우 i=6, m=6, n=93임) 기존의 K-NN 방법은 실시간 단계의 시간복잡도가 O(m*n)이고, Bayesian 방법은 O(i*m*n)인데 반하여 제안하는 결정 트리 방법의 시간복잡도는 O(i*m)이다. 따라서 제안하는 방법이 기존의 방법보다 효율적이다. 본 논문에서 제안하는 결정 트리 방법의 정확도 실험 결과, 평균 오차가 3.241m로 기존 방법과 비교해 봤을 때 오차 면에서는 크게 차이가 나지 않았다.

즉, 결정 트리 방법은 K-NN 방법과 Bayesian 방법에 비해 오차는 비슷하고 효율성은 뛰어나게 좋았다. 향후에는 위치 추정할 범위를 캠퍼스로 확대하고 다른 보정 자료를 이용하여 평균 오차 거리를 줄이는 방법으로 연구하고자 한다.

참고문헌

[1] Harter, A., Hopper, A., Steggles, P., Ward, A., Webster, P., "The Anatomy of a Context- Aware Application," Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking, MOBICOM'99, Seattle, Washington, USA, August 1999, pp.59-68.

[2] Want, A., Hopper, V., Falco and J., Gibbons, "The Active Badge Location System", ACM Transactions on Infomation Systems 10, pp. 91-102, Jan. 1992.

[3] Bahl, P., Venkata, N., Padmanabhan, "RADAR:An in-building RF-based user location and tracking system, INFOCOM, pp. 775-784, Mar. 2000.

[4] Tsung-Nan, L., Po-Chiang, L., "Performance Comparison of Indoor Positioning Techniques based on Location Fingerprinting in Wireless Networks," Proceedings of the 2005 International Conference on Wireless Networks, Communications and Mobile Computing, Volume 2, 13-16 June 2005 pp. 1569 1574.

[5] Ito, S., Kawaguchi, N., "Bayesian based location estimation system using wireless LAN," Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops, 2005. (PerCom 2005 Workshops), 8-12 March 2005, pp. 273 - 278