에이전트 시스템 기반 로봇 설계 및 구현

박기덕[a] 양정진[b]

[a] *가톨릭대학교 컴퓨터공학과*
경기도 부천시 원미구 역곡2동 산43-1
Tel: +82-2-2164-4678, E-mail: beda.park@gmail.com

[b] *가톨릭대학교 컴퓨터공학과*
경기도 부천시 원미구 역곡2동 산43-1
Tel+82-2-2164-4678 E-mail: jungjin@catholic.ac.kr

Abstract

로봇 시스템에서 지적 활동의 중심적 자료구조가 될 공유 메모리를 이용하여 로봇을 이루는 여러 Component Agent가 수집한 Context들과 산출된 Data들을 모아 Mental State를 구축한다. 로봇 Agent를 이루는 모든 컴포넌트들 간의 Data 교류는 Mental State를 통하여 일어나고, Reactive Layer와 Deliberative Layer로 구분 된 로봇을 구성하는 Agent들은 상황에 따라 변화된 context와 data값을 실시간으로 Mental State에 기록, 갱신한다. 이를 통하여 실시간 미션 수행 로봇이 효과적으로 목표를 수행할 수 있는 시스템 구조를 제시하고자 한다. 또한, 이러한 구조를 적용한 자가위치탐지 자율주행 로봇의 구현을 통해 본고에서 제시한 시스템 구조의 실현 가능성을 보이고자 한다.

Keywords:

Robot; Agent system

서 론

로봇이 유비쿼터스 컴퓨팅[1]의 참여자의 일환으로써 이질적이고 분산된 환경에서의 문제 해결과 서비스를 제공하기 위해서는 자신이 처한 환경을 확실히 인지[2]할 필요가 있으며, 이렇게 인지된 정보를 통해 로봇의 행동을 제어한다. 하지만 분산되어있는 유비쿼터스 환경에 속해 있는 로봇이 다양한 Perception Device를 통해서 인지된 Context들과 수집된 Data들을 운용하기 위한 구조적인 방법과 로봇 설계 방식이 필요하게 되었다. 이와 같은 목적을 위해서 Multi Agent System[3]을 이용한 로봇 설계[4]와 Blackboard System[5]을 이용한 로봇 설계[6]와 같이 다양한 시도와 연구가 진행되고 있다. 본고에서는 로봇을 구성하는 다양한 Agent들이 수집하고 산출한 Data들을 효율적으로 저장하고 운용하기 위하여 로봇의 컴포넌트 구조를 제시하고 이를 이용한 자가위치탐지 자율주행로봇을 구현하고자 한다.

관련연구

분산 인공 지능의 한 영역인 블랙보드구조 (Blackboard Architecture)[5]는 분산된 에이전트들이 공동작업을 통하여 문제를 해결하기 위한 방법을 제공한다. 블랙보드구조의 한 요소인 블랙보드는 문제에 적합한 추상화된 몇 개의 레벨로 분할되어 있다. 특정한 레벨을 통해 통신을 수행 하던 에이전트[3]들은 상호작용을 통하여 인접한 레벨로 전이할 수 있다. 이러한 방법을 통해 에이전트들이 수집한 데이터는 한 레벨을 통해 공유되고, 이렇게 공유된 데이터들을 활용하여 목표로 하는 단계로 전이 할 수 있다.
크리켓 시스템[7]은 초음파를 이용한 거리 측정방식, 즉 음파의 비행시간(Time of Flight)의 측정에 의하여 거리를 측정방식을 통하여 미리 저장된 발신기의 위치 정보를 이용하여 현재의 위치를 탐지하는 근접측정 방식이다. 본고에서 이용된 시스템은 크리켓 시스템을 활용하여 발신기로부터 전달된 복수개의 신호 중에 3개의 신호의 음파비행시간을 이용한 삼각측량을 통하여 로봇의 위치를 측정한다.

Software 구성

IDIS Agent는 [그림 1]에 표현 한 것 같이 Qplus Linux Kernel 2.6 운영체제를 기반으로 환경의 변화를 감지하기 위한 Sensor 컴포넌트와 환경을 변화시키기 위한 Actuator 컴포넌트를 기반으로 동작한다. IDIS Agent는 목적을 성취하기 위한 지능적인 개체로 표현 할 수 있으며 Agent의 전체 골격과 각 컴포넌트들을 프로세스로 구현하여

논리적으로 병렬적인 행동 수행이 가능하게 설계 되었다. IDIS Agent를 시스템에 배치하기 위해서는 에이전트의 지능적인 자료구조와 반사적인 컴포넌트들과 지능적 사고를 수행하는 컴포넌트들을 추가 하여 주어야 한다.

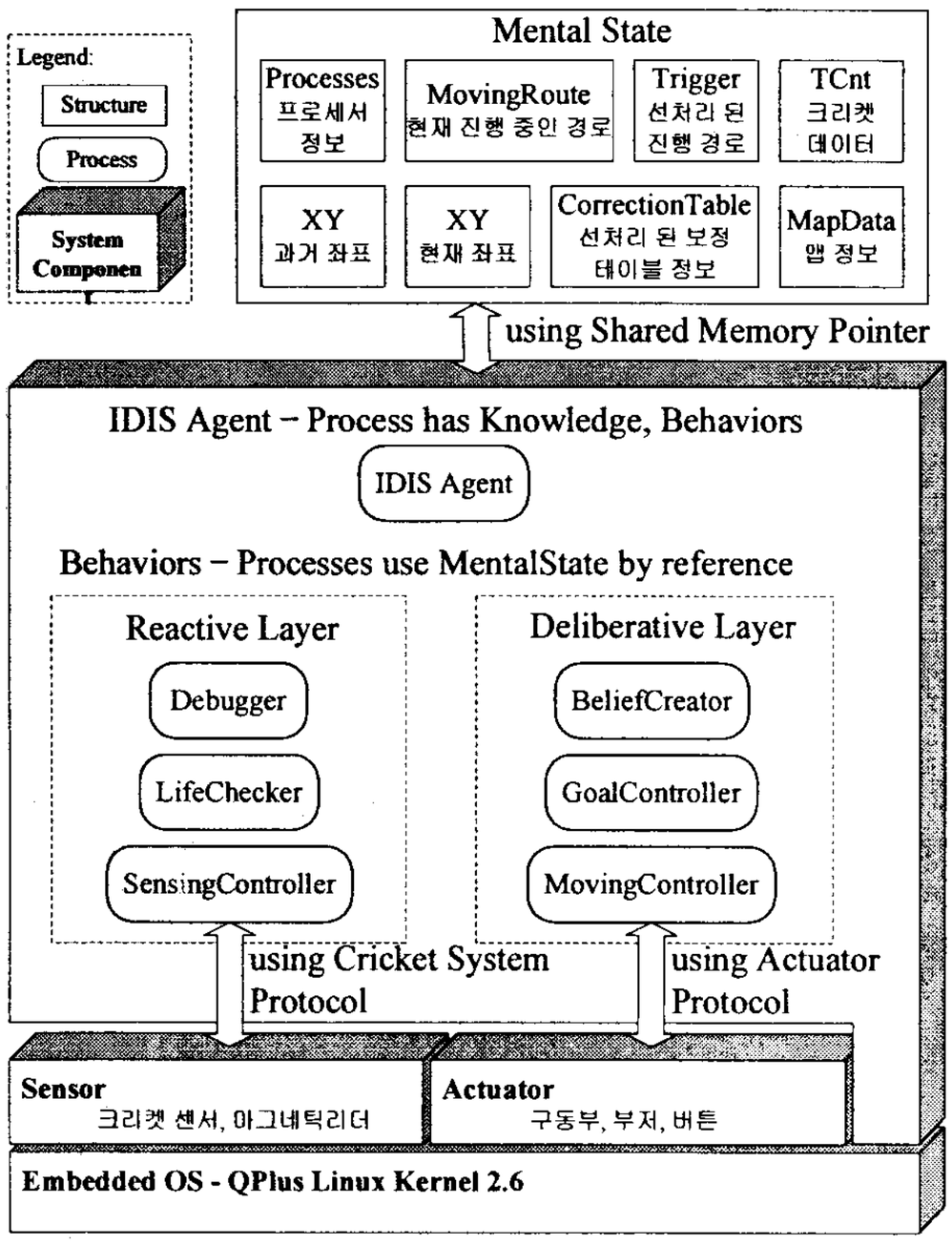

그림 1 - IDIS Agent 시스템 컴포넌트 구조

IDIS Agent는 [그림 1]의 상단에 표현한 Mental State 구조체를 지능적인 자료구조로 사용하며 Mental State는 에이전트의 관점에서 환경을 인식하는 정보와 지식들을 저장한다. Mental State 구조체는 공유 메모리에 할당되어 IDIS Agent를 구성하는 프로세서에서 참조에 의한 접근 방식으로 사용된다. Mental State는 내부적으로 IDIS Agent를 구성하는 프로세서 정보, 위치 정보, 상태 정보, 환경 정보, 선처리된 지식들을 표현한 구조체들의 조합으로 정의되어 있다.

IDIS Agent를 구성하는 컴포넌트들은 [그림 1]의 중간에 표현 하였듯이 크게 두 가지로 분류 된다.

첫 번째는 일정한 주기를 가지고 실행되는 Reactive Layer에 해당하는 컴포넌트들이며 환경 정보와 상태 정보를 주시하면서 에이전트 스스로를 보호하고 환경에 악영향을 끼치는 것을 방지하는 역할을 수행하는 컴포넌트, Mental State의 정보를 디버깅 하기 위한 역할을 수행하는 컴포넌트로 구성된다.

두 번째는 Deliberative Layer에 해당하는 컴포넌트들이며 IDIS Agent의 Mental State를 에이전트의 관점(Belief)으로 초기화 하기 위한 컴포넌트와 에이전트의 목표들을 달성하며 전체 목표를 성취하기 위한 컴포넌트, 현재의 위치와 상태를 고려하여 에이전트의 동작을 수행하는 컴포넌트로 구성된다. [그림 1]의 하단에 표현된 Sensing Controller 컴포넌트와 Movingcontroller 컴포넌트는 Sensor 컴포넌트와 Actuator 컴포넌트와 상호작용 하기 위하여 Cricket System 프로토콜과 Actuator 프로토콜 사용한다. SensingController 컴포넌트는 일정한 주기 (i.e. 크리켓 시스템이 허용할 수 있는 주기)를 가지며 크리켓에서 감지된 위치 정보를 기반으로 보정알고리즘을 통하여 보정된 데이터로 IDIS Agent의 현재 위치를 판별하여 에이전트가 사용하는 위치 정보를 생성하고, 장애물의 감지 유무를 판별하여 에이전트의 자기보호를 수행한다. MovingController는 Mental State에 기록되어 있는 현재 에이전트가 이동해야 하는 경로를 바탕으로 상황에 적합하게 이동을 담당하는 역할을 수행한다.

Agent Based Approach

에이전트는 다중에이전트 시스템(e.g. Jade), 분산 시스템(e.g. CORBA)와 같은 분산 정보시스템과 통신하는 컴포넌트를 추가하여 상황에 알맞게 지식을 확장, 수정하고, 다른 에이전트와 협동하는 컴포넌트를 추가 할 수 있다.

BeliefCreator 컴포넌트는 파일 입출력을 통하여 보정 정보와 선 처리 된 이동경로(장애물은 미리 알려져 있기 때문에 적용)를 MentalState에 업데이트 한다. 응용 사례의 예로 우리가 제안한 시스템은 파일 입출력 대신 TCP/IP기반 통신을 하는 컴포넌트로 교체할 수 있다. Debugger 컴포넌트는 일정한 주기를 가지고 MentalState를 모니터링 한다. TCP/IP기반의 단순한 응용을 제공하면 GUI기반의 데이터 추출과 모니터링이 가능하다. 컴포넌트들을 객체지향, 규칙기반, 스크립트기반의 추상화 된 설계 기법을 적용하면 높은 재사용성을 가지며 다른 에이전트를 구성하는 생산성을 높일 수 있다.

컴포넌트의 흐름

IDIS Agent가 목표를 수행하는 과정은 [그림 2]에 표현한 활동 다이어그램으로 요약할 수 있다. IDIS Agent가 실행되면 첫 번째로 BeliefCreator 컴포넌트를 이용하여 Mental State를 초기화(i.e. 장애물의 위치, Grid World로 재구성된 맵, 각 Goal에 따른 맵 정보) 한다. 그 다음 Reactive Layer의 컴포넌트들을 수행하여 주기적으로 환경의 정보와 상태정보들을 Mental State에 수정한다. SensingController는 크리켓 데이터를 감지하는 도중에 크리켓 데이터를 생성하는 모듈이 중지 할 때 프로세스가 무한 대기 상태가 될 수 있다. LifeChecker는 무한 대기 상태가 된 컴포넌트를

감지하여 만약 무한 대기 상태에 빠져있다면 프로세서를 재 실행 시키는 역할을 수행한다. SensingController는 만약 IDIS Agent가 이동 중에 장애물을 감지하게 되면 이동을 중지하고 장애물 감지 유무를 MentalState에 기록한다.

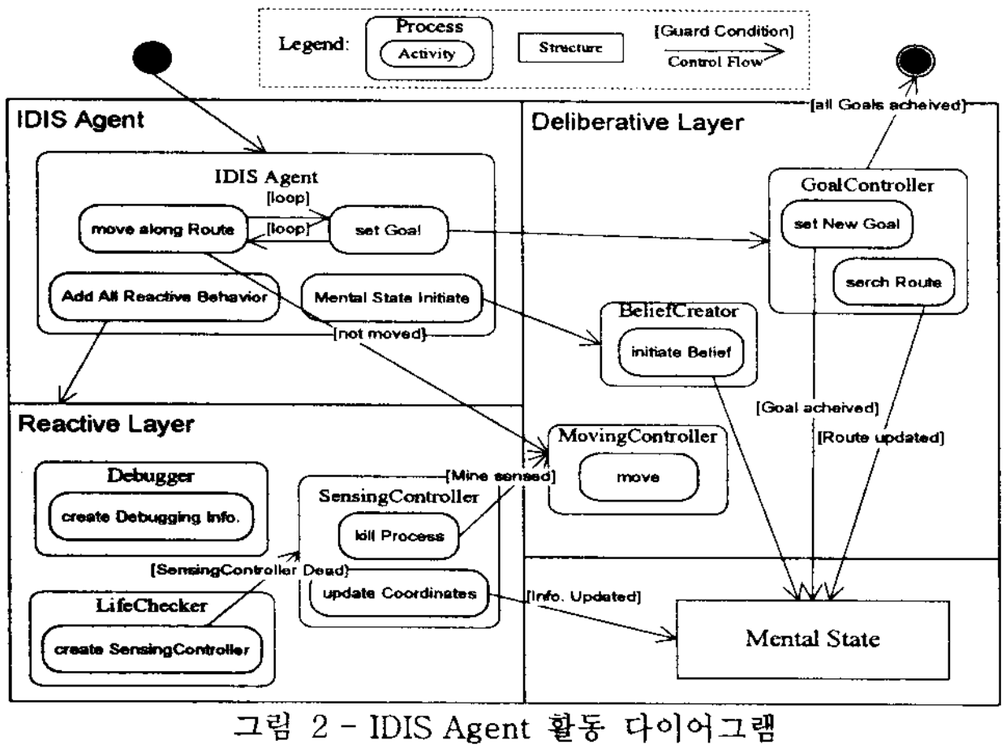

그림 2 - IDIS Agent 활동 다이어그램

IDIS Agent는 GoalController 컴포넌트와 Moving-Controller 컴포넌트를 번갈아 가면서 수행한다. GoalController는 전체 목표를 달성하기 위하여 나누어진 부 목표정보를 이용하여 부 목표를 하나씩 달성하기 위한 정보와 지식을 Mental State에 기록한다. 부 목표를 달성하기 위한 주행 경로와 목표의 달성 유무를 판단하여 추가적인 행동, 다음 부 목표를 설정하기 위한 행동을 수행하고 만약 장애물이 감지된 상황이면 경로를 탐색하여 우회하기 위한 경로를 설정한다. MovingController는 GoalController에서 작성한 목표 달성을 위한 경로와 행동을 참고하여 목표 달성에 필요한 이동을 수행한다.

컴포넌트 인터페이스

컴포넌트-컴포넌트 간의 인터페이스는 각 프로그램 간에 추가적인 오버헤드 발생의 부담 없이 사용 가능하도록 고려하여 설계하고 의존 관계를 최소화하여 분리된 형태를 추구하였다. 앞서 설명한 Mental State가 컴포넌트 간 인터페이스의 대표적인 예가 되겠다. 공유 메모리 접근을 통한 프로그램 제어를 위한 데이터를 보다 빠르고 간편하게 얻어낼 수 있다. 각 컴포넌트는 독립적으로 동작이 가능함과 동시에 Agent를 통하여 각 컴포넌트 간 시그널 교환을 통해 상호작용할 수 있도록 설계되었다.

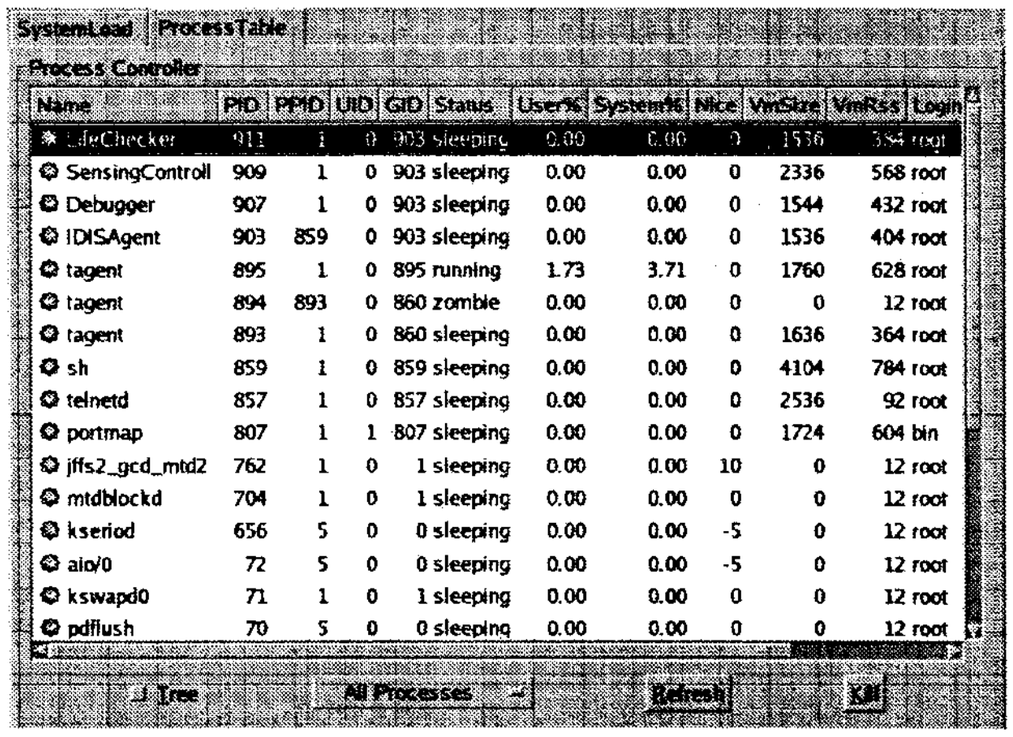

그림 3 - IDISAgent 구동 시 프로세스들의 상태

Results

IDIS Agent는 작은 단위의 컴포넌트(i.e. 프로세스)를 필요한 시점에 적재 하여 실행한다. 따라서 시스템의 모든 기능이 프로세스 이미지 형태로 메모리에 상주 하지 않기 때문에 자원 소모가 적다. 또한 실행 시점에 개선된 프로세스 이미지를 배치하는 것으로 컴포넌트를 개선할 수 있다(단, Mental State의 구조체 형태가 바뀌는 경우는 제외한다). [그림 4]은 Visual Esto에서 제공하는 타겟 시스템 모니터로 IDIS Agent가 작동하는 동안 사용하는 자원을 측정한 것이다.

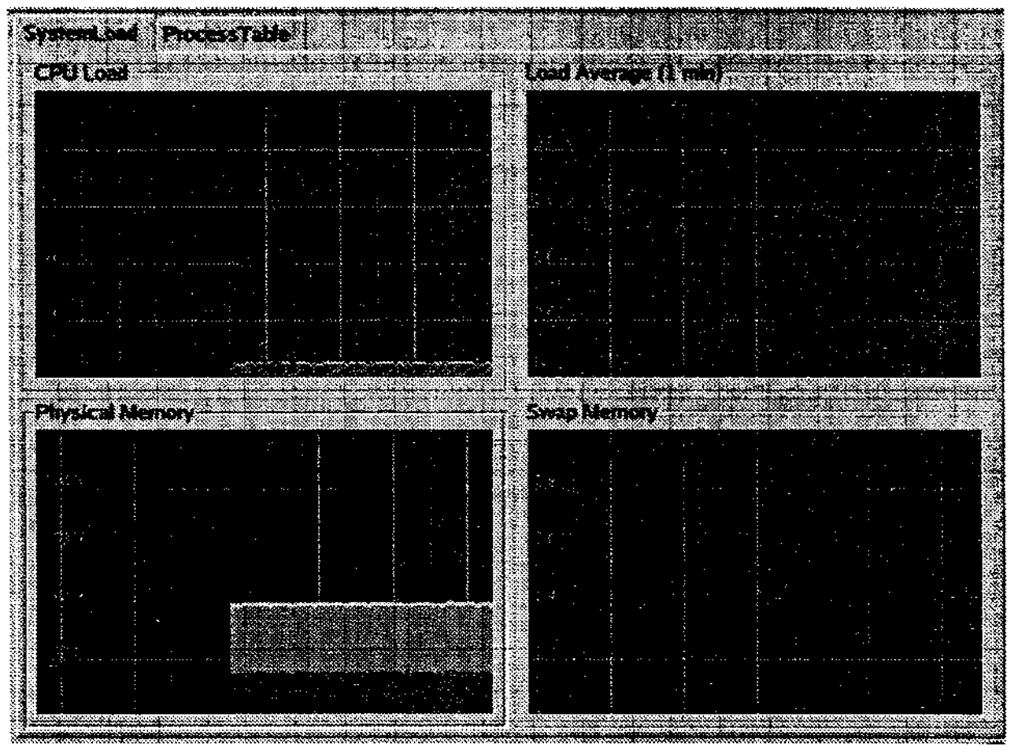

그림 4 - 자원 측정

CPU는 평균 적으로 10% 미만의 낮은 사용률을 보이고 소요되는 메모리는 일정한 크기를 유지하였다. 따라서 남은 자원은 추가적인 컴포넌트, 장비등과 같이 다양한 목적으로 활용 될 수 있다. 실행 시점에 Mental State를 교체하기 위하여 고려할 수 있는 방안으로 경량의 DBMS를 사용하여 데이터와 응용코드를 분리 하는 방법, 다른 형태의 Mental State를 생성하여 동기화 시켜주는 프로세스를 도입하는 방법을 예로 들 수 있다.

후 기

본 연구는 유비쿼터스 자동화 컴퓨팅과 네트워크 프로젝트, 정보통신부(MIC) 21세기 프론티어 R&D 사업의 연구결과로 수행되었음.

참고문헌

[1] Gregory D. Abowd and Elizabeth D. Mynatt (2000) "Charting Past, Present, and Future Research in Ubiquitous Computing," *ACM Transactions on Computer-Human Interaction*, Vol. 7, No. 1.

[2] Harry Chen, Tim Finin, Anupam Joshi. (2004) "An Ontology for Context-Aware Pervasive Computing Environments, " *Journal of Special Issue on Ontologies for Distributed Systems, Knowledge Engineering Review*, pp. 197-207.

[3] Katia P. Sycara (1998) "Multiagent Systems," AL magazine Volume 19, No.2 Intelligent Agents Summer

[4] Bianca Inoocenti, Beatriz Lopez and Joaquirn Salvi (2003) "Multi-Agent System Architecture with Planning for a Mobile Robot," *Proceedings of the 10th Conference of the Spanish Association for Artificial Intelligence, CAEPIA.*

[5] Daniel D. Corkill (1991) "Blackboard Systems," AI Expert 6(9):40-47.

[6] L.Lau, H.Y.K.Lau, Albert Ko (2003) "A Distributed Blackboard-based Control System for Modular Self-Reconfigurable Robots," *Proceedings of the 11th IEEE Mediterranean Conference on Control and Automation,*

[7] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan (2000) "The Cricket Location-Support System," *Proceedings of the 6th annual international conference on Mobile computing and networking*, pp. 32-43.