지능형 멀티 에이전트 로봇시스템을 위한 통신시스템의 설계

Design of Communication System for Intelligent Multi Agent Robot System

김 준 엽, 박 승 민, 고 광 은, 장 인 훈, 심 귀 보*

(Jun-Yeup Kim¹, Seung-Min Park¹, Kwang-Eun Ko¹, In-Hun Jang², and Kwee-Bo Sim¹)

¹Chung-Ang University

²Korea Institute of Industrial Technology

Abstract: In the ad-hoc wireless network environment, that the fixed sensor nodes and the sensor nodes to move are mixed, as the number of the sensor nodes with mobility are getting more, the costs to recover and maintain the whole network will increase more and more. This paper proposed the CDSR (Cost based Dynamic Source Routing) algorithm being motivated from the typical DSR algorithm, that is one of the reactive routing protocol. The *cost function* is defined through measuring the cost which any sensor node pays to participate in the whole network for communication. It is also showed in this paper that the proposed routing algorithm will increase the efficiency and life of whole sensor network through a series of experiments.

Keywords: sensor network, Ad-hoc routing protocol, MARS (Multi Agent Robotic System), ZigBee module

I. 서론

멀티 에이전트 시스템에 속해있는 지능형 에이전트들은 서로 통신하고 상호작용 해야만 한다. 그리고 이를 구현하기 위한 공통된 언어가 필요한데 이를 ACL (Agent Communication Language)라고 하며 ACL 개발을 위한 많은 연구가 진행되어 왔다. 멀티 에이전트 시스템에 있어서 가장 널리 사용되는 ACL로서는 KQML [1]과 FIPA-ACL [2]이 있다. 이러한 언어들은 주로 관리기관의 decision making [3,4], 재정관리[5], 항공기 유지보수[6] 등에 사용되는 software agent들을 위해서는 매우 유용하게 사용되어 왔다. 그러나 에이전트 간 협상(negotiation)을 위해 고수준의 정보를 주고받기 위한 통신요구에는 기존의 ACL이 유용할지 모르나, 저수준의 데이터를 주고받는 시스템, 즉 시간적인 제약이나 대역폭(bandwidth)의 제한을 가진 시스템에서 기존의 ACL은 상당히 불리한 오버헤드를 가진다[7].

로봇시스템의 경우가 이러한 범주에 속하는 전형적인 예라 할 수 있다. 이러한 시스템들은 원격신호, 비디오, 오디오 그리고 센서로부터의 측정 데이터 등을 실시간으로 매우 빈번히 전송하면서 그러나 상대적으로 느린 무선 네트워크 또는 RF 모듈 등을 이용한다. 실제 로봇 시스템들 사이에 주고받는 데이터는 원격 명령어 같은 비교적 단문의 메시지들이거나 스트리밍 비디오 같은 큰 용량의 멀티미디어

어 파일인 경우가 많다. ACL로 인해 발생하는 오버헤드는 단문의 메시지인 경우 가장 큰 문제이기도 하지만 멀티미디어 파일을 주고받기에도 ACL은 적절한 언어는 아니다 [7]. 따라서 본 논문에서는 기존의 ACL을 사용하면서 단문의 메시지를 주고받기 위한 통신방식을 추가한 이중 구조의 통신 방식을 제안한다.

II. 지능형 멀티에이전트 시스템의 통신 시스템 구성

본 논문에서 지능형 멀티에이전트 시스템의 작업환경은 각종 센서 및 기기와 유 무선으로 연결되어 센서네트워크를 이루고 있음을 가정하였으며 사람, 에이전트, 그리고 기기들 사이에 일어날 수 있는 communication 형태를 그림 1과 같이 4가지 형태로 구분하였다. 그림 1의 “a”에서는 인간과 지능형 에이전트 사이의 communication을 기술하며 자연어 처리(natural language process), 음성인식(voice recognition), 영상인식(image processing) 등을 통한 HCI (Human-Computer-Interaction) 기능을 요구한다. “b”는 지능형 에이전트간의 통신(interagent communication)을 기술하며 저수준의 통신과 고수준의 통신을 함께 이용하는 이중구조의 통신방식을 가진다. “a”와 “b”에서는 사용자의 의도를 기술(intentional stance description)하기위해 어떤 상태에 대한 정보를 믿음, 욕구, 희망 등의 정신적인 상태 또는 태도로 표현할 수 있는 기능을 포함한다. “c”의 경우는 통상적인 홈네트워크 서버와 end device들 간에 요구되는 수준의 통신프로토콜이 요구되며, 센서 네트워크상에 연결된 기기들로부터 센서 데이터들을 받아들이고 이러한 데이터들을 융합하여 상황 인식(context awareness)을 할 수 있는 기능을 포함한다. “d”는 버튼, LED/LCD, sound 등의 기기자원을 통한 사용자와의 직접적인 인터페이스를 의미하며 기기제어를 위한 제어명령이 사용된다.

* 책임저자(Corresponding Author)

논문접수: 2012. 5. 15., 수정: 2012. 6. 14., 채택확정: 2012. 6. 30.

김준엽, 박승민, 고광은: 중앙대학교 전자전기공학부

(jy915@cau.ac.kr/sminpark@cau.ac.kr/kkeun@cau.ac.kr)

장인훈: 한국생산기술연구원 선임연구원(inhuns@kitech.re.kr)

심귀보: 중앙대학교 전자전기공학부 교수(kbsim@cau.ac.kr)

※ 본 논문은 한국연구재단 중견연구지원사업(No. 2011-0029861)에서 지원하여 연구하였으며 연구비 지원에 감사드립니다.

그림 1. Open 환경에서 사람, agents, 기기들 사이의 communication 유형.

Fig. 1. The communication types among human, agents and devices.

그림 1에서 정의한 4가지 통신 형태를 실제로 구현하기 위해서 다음과 같이 H/W를 설계하였다. 스마트 홈(smart home) 시스템과 같은 고정형 에이전트의 경우 유선기기들과는 RS485통신을, 무선기기들과는 IEEE 802.15.4 기반의 지그비(Zigbee) 프로토콜을 사용하여 네트워크를 형성하도록 하였다. 이동 로봇과 같은 모바일 에이전트는 지그비를 이용해 다른 에이전트 및 기기들과 통신하도록 하였는데 이 경우 이동에 따른 네트워크를 스스로 구성(self-organization)할 수 있어서 라우팅 경로를 유지할 수 있는 기능을 요구한다. 또한 지능형 에이전트들은 인간과 에이전트, 에이전트와 에이전트 사이의 고수준의 통신프로토콜 구현이나 대용량 정보전달을 위해서는 802.11b 기반의 무선 랜을 통한 네트워크를 형성하도록 하였다.

III. Cost 기반 Ad-hoc 센서 네트워크 라우팅 알고리즘

1. 최소거리 기반의 라우팅 알고리즘

1.1 Ad-hoc 무선 네트워크 환경과 Routing Protocol

센서 네트워크의 주요한 특징으로는 센서 네트워크를 구성하는 각각의 센서 노드들의 수가 수십에서부터 수백, 수천 이상의 대규모로 구성될 수 있고 배터리를 이용하기 때문에 저 전력으로 작동되어야 한다는 것이다. 그리고 multi hopping 방식으로 통신을 하며 센서 노드들의 이동 또는 fail 등으로 인해 네트워크 토폴로지가 빈번히 변한다는 것 등을 주요한 특징으로 들 수 있다[8,9]. 이러한 특징들 가운데 특히 전원 소비를 최소화 하기 위한 노력들이 많이 있어 왔는데 전통적인 data aggregation, join, average와 같이 데이터 자체에 대한 통계처리를 통해 무선 송수신 횟수를 최소화 함으로써 전원소비를 줄이려는 시도가 많이 있어왔다[9,10]. 그러나 그 적용에 따라서는 데이터 자체에 대한

통계적인 프로세싱이 불가능한 경우가 있다. 대표적인 예로 메디컬 방면에 응용이 되어 환자나 일반인의 건강상태를 감지하는 것이 각 센서 노드들의 역할인 경우 사용자들의 데이터를 통계 처리할 수는 없으며 개별적으로 관리가 되어야 한다.

또 다른 방법으로 selection 순서의 재정의, 그리고 위치 정보를 이용한 선택적인 query 등과 같은 query optimization을 통해 네트워크 전체의 전원소비를 줄이려는 시도가 제안되었으며[9], 위치정보를 가지고 지도검색과 같은 공간 데이터베이스의 검색에 널리 이용되는 R-Tree 기반의 Routing Tree 구성[10] 등이 연구되어 있다. 또한 위치정보와 각 센서 노드의 센싱 범위, 통신 반경 등에 관한 정보를 이용하여 사전에 센서 노드들을 잘 배치시키면 네트워크 자원을 좀 더 효율적으로 사용할 수 있기 때문에 센서의 효율적인 배치와 관련된 연구 또한 있어왔다[12,13].

본 논문에서는 고정된 센서 노드들과 이동하는 센서 노드들이 혼재되어 있는 Ad-hoc 무선 네트워크 환경을 가정한다. 이러한 환경에서 다단계 호핑 방식(multi hopping)의 센서 네트워크 통신을 하는 경우 이동성을 가진 센서 노드들이 많아질수록 네트워크의 경로를 재구성하고 유지하는데 드는 비용은 점점 더 많아진다. 기존의 Ad-hoc routing protocol은 그림 2와 같이 세 종류로 나눌 수 있는데, Proactive routing protocol(table-driven)의 대표적인 방법이 DSDV (Destination-Sequenced Distance-Vector) Routing [14]이고 Reactive routing protocol(Demand-driven)의 대표적인 방법이 DSDV 기반의 AODV (Ad-hoc On-Demand Distance-Vector) [15]와 DSR (Dynamic Source Routing) [16,17]이다.

Ad-hoc routing protocol의 근간은 거리 기반의 Distance vector algorithms으로서 두 센서 노드간의 경로는 각 센서 노드 사이의 거리의 합이 최소가 되는 경로로 정의한다. 각 노드는 목적지 노드에 대한 리스트, 목적지로 가기 위해 될 다음 노드의 주소, 목적지까지의 hopping count 그리고 중복된 통신요청(looping problem)을 회피하기 위한 sequence number를 항목으로 하는 routing table을 가지고 있다. Proactive routing protocol은 주기적으로 자신의 routing table을 이웃노드들에게 전달하여 전체적으로 최신의 routing table을 유지함으로써 각 목적지까지의 최단거리로 갈 수 있는 경로를 가지게 된다. 그러나 이러한 경우에 필연적으로 필요 없는 통신 traffic이 발생하게 된다.

Proactive routing protocol과 반대되는 개념으로 필요시에만 라우팅 경로를 만드는 방식을 Reactive routing protocol이

그림 2. Ad Hoc 라우팅 프로토콜의 종류.

Fig. 2. Ad Hoc routing protocols.

라 하며 대표적인 방법으로는 AODV와 DSR이 있다. 본 논문에서는 DSR 프로토콜을 기본으로 상기의 routing table을 가지지만 거리 값이 아니라 센서 노드 자신의 라우팅 비용을 나타내는 척도를 사용하여 routing 경로를 설정하는 방법을 제안한다.

1.2 DSR 알고리즘에서의 Route Discovery

DSR에서 메시지를 전송할 경우 그림 3과 같은 Route request (RREQ)를 네트워크상에 broadcasting하는 source routing을 통해 라우팅 경로를 찾게 되는데 패킷 헤더에 목적 노드에 도달할 때까지의 지나간 노드를 명시하여 전송한다.

여기서, RREQ는 Route request, sender는 패킷 전달자(이전 노드), dest.는 목적지 노드, hop는 패킷이 전달된 횟수, init.는 시작 노드, req. ID는 패킷 구분자를 각각 나타낸다.

그림 4의 노드1(N1)은 노드8(N8)을 목적지로 하는 Route request(RREQ) 패킷을 그림 5의 예와 같이 전체 네트워크에 브로드캐스팅 한다.

N2는 RREQ 목적지가 자신이 아니기 때문에 RREQ를 다시 브로드캐스팅 한다. 이 때 자신의 라우팅 캐시에 패킷에 대한 정보를 그림 6-6과 같이 기록한다. 그림 6-6 패킷의 Source(S)는 N1, 이웃노드(N)는 N1, 그리고 hopping 수(H)는 1임을 의미한다.

브로드 캐스팅된 패킷은 N3도 받게 되며 N2와 마찬가지로의 작업을 한다. N5는 N2와 N4로부터 브로드 캐스팅된 패킷을 각각 받으며 자신의 캐시를 그림 7과 같이 유지한다. N1에서

RREQ	dest	init	sender	hop	req. ID
------	------	------	--------	-----	---------

그림 3. Route request를 위한 broadcasting packet.

Fig. 3. Broadcasting packet for route request.

그림 4. DSDV의 Table을 가지는 DSR.

Fig. 4. DSR with the Table of DSDV.

RREQ	N8	N1	N1	1	req. ID
------	----	----	----	---	---------

그림 5. N1에서 N8까지의 routing discovery를 위한 broadcasting packet.

Fig. 5. Broadcasting packet for routing discovery from N1 to N8.

D(S)	N	H
N1	N1	1

그림 6. N2의 routing cache.

Fig. 6. Routing cache of N2.

D(S)	N	H
N2	N2	1
N1	N2	2
N4	N4	1
N1	N4	3

그림 7. N5의 routing cache.

Fig. 7. Routing cache of N5.

그림 8. 거리(hopping count)에 의한 라우팅 경로.

Fig. 8. Routing path based on hopping count.

N5까지의 경로를 선택하는데 있어서 N5의 이웃노드(N)는 N2와 N4이며 N2를 거쳐 N1까지 가기위한 hopping count(H)는 2이고, N4를 거쳐 N1까지 가기위한 H는 3임을 의미한다. 따라서 H가 적은 N1-N2-N5가 그 경로로 선택된다.

마찬가지 방법으로 N1에서 N8까지의 경로는 가장 적은 hop count를 가지는 N1-N2-N5-N8로서 선택된다.

최종적으로 N8은 RREQ 패킷의 목적지가 자신이기 때문에 Route Reply 신호를 ACK. 패킷에 담아 보내며 이는 경로의 역순으로 N1에 전달된다. 그림 8은 N8-N5-N2-N1로 전달되는 ACK. 패킷을 나타내고 있다.

2. Cost기반 라우팅 알고리즘

2.1 라우팅 비용 함수의 정의

기존의 DSR 알고리즘에서는 그림 8과 같이 hopping count를 기준으로 N1-N2-N5-N8로의 라우팅 경로가 생성되지만 본 절에서는 네트워크상에서의 라우팅 비용 정보를 추가하여 네트워크 상황에 따라서는 N1-N3-N4-N5-N8이 최적 경로로 선택될 수도 있는 알고리즘을 제안한다.

예를 들어 Ad-hoc 네트워크 환경에서 N3은 고정되어 있지만 N2는 이동 로봇과 같이 빈번히 이동한다면 이 경로는 N2의 이동에 의해 link broken이 발생하여 오래가지 못해 다시 route request를 전 네트워크에 broadcasting해야만 할 것이다. 이처럼 mobility가 큰 노드는 가급적이면 경로선택에 있어서 제외하는 것이 route request를 줄일 수 있기 때문에 네트워크 전체적으로 더 나을 수 있다는 가정을 전제로 한다.

또 다른 예로 우리가 자동차로 목적지까지 가는 경우에 있어서, 거리는 짧지만 차가 많아서 막히는 길과 먼 거리를 돌아가지만 막히지 않아서 시간을 절약할 수 있는 외곽도로가 있는 경우 종종 후자의 길을 택한다. 마찬가지로 네트워크상에 있는 어떤 센서 노드가 거리상으로는 가까울 지라도 병목지점에 있다면 이를 피하여 라우팅 경로를 선택하는 것이 더 바람직할 수 있다.

이러한 개념들을 반영하는 최적의 경로선택 알고리즘을 제안하기 위해 다음과 같은 라우팅 비용 함수 P_i 를 정의한

다. i 번째 센서 노드를 SN_i 로 표시하고, 자신이 네트워크에 참여하여 통신을 위해 지불해야 하는 비용을 P_i 라고 하면, P_i 는 식 (1)과 같이 나타낸다.

$$P_i = w_{i1} \times Power_i + w_{i2} \times Mobility_i + w_{i3} \times Freq_i \cdots \quad (1)$$

여기서 w_{ij} 는 가중치이며, $Power_i$ 는 일반적인 센서네트워크에서 가장 큰 관심사가 되는 센서 노드 SN_i 의 전원관련 정보로서 이미 소모된 전원량 또는 남아 있는 전원량의 역수로 정의할 수 있다. 자신의 power가 일정 수준 이하로 떨어지면 자신과 직접적인 관련이 있는 송수신은 유지하면서 단순 라우팅을 위한 송신은 off함으로써 다소 이기적인 통신을 할 필요가 있다. 또한 power의 감소율을 통해 그 노드의 부하정도를 알 수 있으며 이를 통해 부하가 적은 쪽으로 분산시킨다. $Freq_i$ 는 센서 노드 SN_i 가 네트워크에 참여하는 빈도로서 power의 감소율로 대신할 수 있다. 이 값이 클수록 네트워크 충돌이나 traffic 부하가 클 확률이 높다. 즉 상술한 ‘막히는 길’이 되는 것이다. $Mobility_i$ 는 SN_i 의 이동성을 나타내며 이 값이 클수록 네트워크상의 전출입이 빈번하므로 라우팅 경로로서는 신뢰성이 떨어지게 된다. 이동 로봇과 같이 움직임이 심한 노드를 통한 경로보다는 움직임이 없는 노드를 통한 통신경로 설정을 택함으로써 link broken에 의한 route recovery를 줄이고 보다 안정적인 통신을 할 수 있다. 그 밖에 통신 에러의 발생빈도 또한 고려할 수 있다. SN_i 에서 n 개의 센서 노드들을 거쳐 SN_j 에 도달하는 경로에 있어서 예상되는 누적된 총 라우팅 비용 C_{ij} 는 식 (2)와 같이 구할 수 있다.

$$C_{ij} = P_i + P_1 + P_2 + \cdots + P_{n-1} + P_n + P_j \quad (2)$$

여기서 Cost C_{ij} 는 source에서 현재 노드까지 누적된 총 라우팅 비용, P_i 는 자신이 네트워크에 참여하여 통신을 위해 지불해야 하는 비용을 나타낸다.

2.2 Cost함수에 기반을 둔 Route Discovery

라우팅 비용(Cost) 정보를 이용하여 이 값을 최소로 하는 경로를 택한다면 그림 10과 같은 라우팅 경로가 생성될 수도 있다. 여기서 C_{ij} 는 양의 값으로 계속 더해지는 값이기 때문에 노드를 많이 거칠수록 즉, 거리가 멀수록 커지므로 기존의 Hopping Count값을 반영한다고 할 수 있다.

그림 10에서 N2는 전원 소비가 많고 이동성이 많은 센서 노드로서 전원소비가 많다는 것은 얼마 가지 못해 전원을 모두 소비하여 그 기능을 상실할 수 있다는 것을 의미한다. 본 논문에서 주장하는 바는, 센서 노드의 기본적인 역할은 자신의 센서를 이용하여 데이터를 수집하고 그 데이터를 어떤 base station으로 보내는 것이며 라우팅 기능은 이차적인 기능이라고 할 수 있기 때문에 자신의 전

FREQ	dest.	source	sender	Cost	Pi	req. ID
------	-------	--------	--------	------	----	---------

그림 9. 라우팅 비용을 고려한 Route request broadcasting packet.
Fig. 9. Route request broadcasting packet considering routing cost.

원소비가 많은 경우 라우팅 기능을 줄이거나 정지함으로써 그 수명을 연장하여 자신의 기본 기능인 센싱 기능을 수행하여야 한다는 것이다. 전체 네트워크를 위한 라우터로서의 기여와 자신의 수명 사이에서 라우팅 비용함수는 적절한 기준을 제시한다고 볼 수 있다.

3. Route Maintenance

Route Maintenance는 기본적으로 III.2.2절의 Route discovery에서 찾아낸 경로를 잘 보관하고 유지하는 것에 관한 내용이다. 이를 위해서 각 센서 노드들은 메모리에 학습한 경로를 저장하고 최신의 정보로 계속 업데이트해야 하며 때로는 캐시를 사용하여야 한다.

경로 상의 link broken이 발생하는 경우에는 RERR (route error) 메시지를 최초의 노드로 전달하며 RERR을 수신한 최초노드는 다른 우회 루트가 있을 경우 이를 이용하여 데이터 전달을 계속하고 그렇지 않을 경우 route discovery를 다시 수행해야한다.

일반적으로 센서 노드들의 하드웨어 사양 가운데 캐시나 메모리 크기는 그다지 크지 않기 때문에, 학습한 라우팅 경로를 저장하는 양은 센서 네트워크의 크기가 커질수록 제한적일 수밖에 없다. 이 경우 메모리 및 캐시의 일반적인 최적화 알고리즘이 수행될 수 있다.

3.1 라우팅 비용(cost) 정보 C_{ij} 의 갱신

C_{ij} 를 계산하기위한 변수들, 즉 전원과 네트워크에의 참여도 등은 네트워크 환경이나 특정 목적에 따라 시간이 지나면서 계속해서 변하는 값들이다. 그러므로 지속적으로 이

그림 10. Cost함수에 기반을 둔 Route Discovery.

Fig. 10. Route Discovery based on Cost function.

그림 11. Cost함수에 기반을 둔 최종 라우팅 경로 설정.

Fig. 11. Final routing path based on Cost function.

러한 값들을 갱신해주어 최신의 값으로 유지할 필요가 있다. Dynamic하게 변하는 네트워크 환경에 더 잘 적응하기 위해서 구해진 라우팅 경로 또한 수명(life time)을 가질 필요가 있다.

그림 11에서 볼 수 있듯이 C_{ij} 는 메시지의 송수신이 이루어지는 것과 동시에 동적으로 계속 최신의 값으로 유지된다. 이를 위해서 각 센서 노드들은 주기적으로 자신의 P_i 를 계산하고 메시지 전송 시 그 정보를 실어 보낸다.

3.2 Link Broken 발생과 경로 재구성

기본적으로 무선 네트워크는 유선에 의한 네트워크보다 통신신뢰가 떨어지는데, 센서 네트워크상에 각 노드들은 전원문제에 의한 fail 및 이동 등에 의해서 빈번히 link broken이 발생한다. Link broken의 발생뿐만 아니라 packet을 잃어버리거나 packet이 충돌을 일으켰을 때 이를 빠르게 찾아내고 재전송하기 위해 많은 경우 data link 레벨에서 hop-by-hop acknowledgement를 사용한다[17]. 이러한 경우, 통신루트의 유지보수는 패킷을 전송한 노드가 그 자리에서(at that hop) 다음 단계의 센서 노드(hop)가 제대로 작동하고 있는지 아닌지를 결정함으로써 이루어진다.

라우팅 에러가 발생하면 RERR (route error) 메시지를 생성한다. 예를 들어 link broken에 의한 RERR가 발생하면 그림 13과 같은 제한적인 local route discovery를 수행하기도 하는데, 주변 노드들의 루트 캐시 또는 메모리에 목적지까지의 경로가 있는지를 확인한다. 만약 주변 노드들로부터 목적지까지의 경로 정보를 얻지 못한다면 결국 최초의 노

그림 12. 최적 라우팅 경로를 통한 메시지 전송.

Fig. 12. Message transmission through the optimum routing path.

그림 13. 지역적인 경로 복구.

Fig. 13. Local Route Recovery.

그림 14. 복구된 경로를 통한 메시지 전송.

Fig. 14. Message transmission via recovering routing path.

드로 RERR 메시지가 전달되며 최초의 노드는 3.2.2.절의 Route Discovery를 통해 경로를 다시 설정 한다.

라우트 에러 패킷은 에러를 유발시킨 노드의 주소를 포함하고 있어서 최초의 노드로 RERR 메시지를 전달하는 과정에서 그 경로 상에 있는 모든 노드들의 루트 캐시 또는 경로를 저장한 메모리로부터 링크오류를 일으킨 노드의 정보를 삭제하게 된다.

만약 그림 14와 같이 주변 노드들의 메모리에 목적지까지의 경로 정보가 기록되어 있다면 그 경로를 새로운 경로로 하여 메시지 전송을 계속 수행한다. 이 경우 메시지 송수신 과정을 통해 새로운 경로 상에 있는 모든 센서 노드들은 경로와 그에 따른 라우팅 비용 C_{ij} 를 갱신할 것이다.

3.3 Local Route Discovery

일반적으로 Route Discovery를 통해 각 노드에 학습된 네트워크 루트 정보는 시간이 지날수록 축적되며 어떤 두 노드 사이의 경로가 깨질 경우 그 주변 노드는 두 노드 사이를 잇는 또 다른 경로에 대한 정보를 가질 확률이 커진다. 그림 15는 S-A-B-C-D-R로 이루어지는 경로가 C에 의해 깨지면서 B의 이웃 노드 E의 메모리에 학습되어 있던 R로의 경로를 통해 S에서 R로의 경로가 복구되는 경우를 보이고 있다. 그림 15에서 B의 통신 반경 내에 있는 E는 R까지 가는데 있어서 B와는 다른 경로를 가지고 있음을 알 수 있으며 이를 통해 S-A-B-E-F-D-R의 새로운 경로가 생성됨을 볼 수 있다.

이러한 경우를 가정하여 본 논문에서는 다음과 같은 새로운 local route discovery를 제안한다.

- 그림 13은 노드4에서 노드5로 패킷을 전송할 때 노드6과 노드7이 노드4의 통신 반경에 있는 경우로 노드5에 의해 link broken이 발생한 경우를 가정한 예이다. 이러한 경우, 노드4에서 노드5로 패킷을 전송할 때 노드6과 노드7은 이를 엿본다.

- 노드6과 노드7의 라우팅 저장 메모리에 노드4로부터 전송되는 패킷의 목적지가 있는 경우 이의 계속적인 엿보기를 위한 쓰레드를 생성시킨다.

- 일정 시간이 지나 link broken error가 발생했다면 노드6과 노드7은 자신의 메모리에서 노드5에 대한 라우팅 정보

그림 15. 이웃노드에 의한 Route Recovery.

Fig. 15. Route Recovery by neighborhood node.

를 삭제한다.

- 노드6과 노드7은 자신의 메모리에서 목적지까지의 비용 정보를 포함한 link 복구용 패킷을 노드4로 전송한다.
- 노드4는 특별한 local route discovery를 주변 노드에 전송하지 않고서도 주변 노드로부터 새로운 경로에 대한 정보를 얻고 경로를 갱신한다.

제안된 local route discovery는 기본적으로 송수신에 소비되는 네트워크 비용보다는 노드 내에서 이루어지는 프로세스 비용이 훨씬 적다는 것을 가정하고 있으며, 이웃 노드의 수가 이웃 노드를 엮보는 프로세스의 부하가 지나치지 않을 정도란 것을 가정하고 있다.

4. Power Awareness

Power awareness문제는 센서 네트워크 문제에 있어서 가장 기본적이고 중요한 문제로서 지금까지 매우 많은 연구가 있어 왔다. 본 논문에서 각 센서 노드는 자신의 현재 전원상태를 계속적으로 모니터링하며 잔류전원이 적어질수록 라우팅 비용을 점점 크게 하여 네트워크상에서 라우터로서의 역할을 줄이도록 한다. 또한 전원상태가 기준 값 이하로 떨어지면 BAT. LOW 이벤트를 발생시키며, 이 이벤트가 발생하면 다른 노드들의 데이터를 단순 라우팅 하는 라우터로서의 역할은 하지 않고 자신의 센싱 데이터만을 송신하는 이기적인 송수신 전략을 세우게 된다. 또한 단위시간 당 전원의 감소율을 계산하여 네트워크 내에서 routing에 참여하는 빈도를 예측할 수 있으며 이 값을 traffic 부하로 생각할 수 있다. 결과적으로 이 값을 라우팅 비용 함수의 파라메타로 이용하면 traffic 부하를 분산시킬 수 있는 라우팅 경로를 생성시킬 수 있다.

IV. 실험환경 구성

1. 에이전트들과 기기들로 이루어진 네트워크 설계

본 논문에서 가정한 멀티 에이전트의 작업환경은 다수의 지능형 모바일 로봇과 고정된 지능형 시스템을 physical 에

이전트로 하며 여러 종류의 센서, 여러 종류의 단순 작동 기기들이 복합적으로 존재하는 인간 생활환경이다.

지능형 에이전트들 사이에 고수준의 통신프로토콜 구현이나 대용량의 정보전달을 위해서 IEEE 802.11b 기반의 무선 랜을 이용한 네트워크를 형성하도록 하였지만 그보다 훨씬 더 빈번하게 발생하는 저수준의 통신프로토콜 구현과 단문 메시지를 주고받기 위해서는 IEEE 802.15.4 기반의 zigbee 프로토콜을 사용하였다. 특히 이동성을 가진 다수의 로봇이 포함된 경우 네트워크 self-organization 문제와 빈번한 라우팅 경로의 변경에 대응하기 위해서는 저전력의 zigbee 프로토콜이 효과적이다. 또한 각종 센서와 단순 작동 기기에도 상대적으로 저가이고 저전력인 zigbee 프로토콜이 효과적이다.

따라서 본 논문의 실험을 위해서 zigbee 모듈을 직접 설계하고 제작하였다.

2. ZigBee 모듈 설계 및 제작

ZigBee 모듈 설계를 위해 사용한 MG2455는 국내업체인 Radio pulse社에서 개발하였으며 IEEE802.15.4와, ZigBee 표준 스택을 따르고 있는 single-chip solution이다[Mango2009].

MG2455는 2.4GHz RF 트랜시버, 베이스밴드 모뎀, 하드웨어 MAC, 8051 MCU 그리고 내장 Flash 메모리로 구성되어 있으며, 타이머와 UART와 같은 주변 회로, 그리고 일반적인 용도의 I/O 핀들로 구성되어 있다.

그림 16은 MG2455의 기본적인 사양을 나타내며, ZigBee 모듈의 개발환경은 표 1과 같고, 이렇게 개발한 ZigBee 모듈을 그림 17에 나타낸다.

그림 16. MG2455의 기본적인 사양.

Fig. 16. MG2455 Specification.

표 1. 지그비 모듈의 개발환경.

Table 1. Development environment of ZigBee module.

Item	Description
CPU	Radio pulse社 MG2455
PHY & MAC layer	IEEE 802.15.4
ZigBee Stack version	ZigBee Alliance ZigBee Pro
ZigBee Stack Library	Radio pulse社 zPULSE
Compiler	Keil μVision3 8051 C compiler

그림 17. ZigBee 모듈 및 작동 테스트.
Fig. 17. ZigBee module and working test.

V. 실험결과 및 결론

라우팅 비용함수를 이용한 라우팅 알고리즘의 타당성을 보이기 위해 그림 18과 같은 모니터링 프로그램을 설계하였다. 그림 19는 실험을 위해 사용된 센서 노드들의 맵이다. 시뮬레이션을 통해 hop 카운트만을 사용한 라우팅 방법과 cost function을 이용한 라우팅 방법을 비교하였으며 이를 위해 다음과 같은 가정을 하였다.

가정 1: 모든 노드들의 초기 power는 1000으로 동일하게 주었다.

가정 2: 모든 노드들은 10초마다 한 번씩 센서 데이터를 베이스 스테이션으로 전송해야한다.

가정 3: 한번 송신할 때 power 0.5를 사용한다.

가정 4: 센싱 작업과 수신시 소요되는 power는 모두 같다고 가정한다. 따라서 결과 비교에 영향을 미치지 않기 때문에 계산에 고려하지 않는다.

가정 5: 자신의 통신비용을 다음과 같이 가정한다.

$$P_i = 10 - 0.01 \times Power_i + 0.5 \times Freq_i$$

가정 6: mobile agent로 인해 link broken이 늘어나는 것은 너무나 자명하기 때문에 계산의 편의를 위해 본 실험에서는 고려하지 않았다.

그림 20과 21의 (a)는 각 알고리즘에 의해 설정된 초기 라우팅 경로를 나타낸다. 그리고 (b)~(d)는 전원을 모두 소모한 노드들에 의해 발생한 link broken에 대해 각 알고리즘을 이용하여 복구된 라우팅 경로를 나타내고 있다.

그림 22는 각 알고리즘에 대해 센서 노드 N1~N11의 전원 소비에 대한 추이를 나타내고 있으며, 표 2는 Link broken 발생한 시점을 나타내고 있다.

3번째 Link broken이 발생할 때 까지를 보면 제안한 알고리즘의 경우가 기존 알고리즘보다 좀 더 효율적으로 전원을 소비하고 있음을 그림 22와 표 2를 통해 확인할 수 있다. 단, 기존의 알고리즘의 경우 좀 더 오래 살아남는 노

그림 18. Route Discovery 시뮬레이션 프로그램.
Fig. 18. Simulation program for route discovery.

그림 19. 실험에 사용한 센서 노드 맵.
Fig. 19. Sensor node map for experiments.

드들이 있어 Base station으로 센서 데이터를 전송하고 있음을 볼 수 있다. 그러나 그림 20(d)에서 볼 수 있듯 맵을 기준으로 볼 때 지리적으로 치우쳐 있어 전체를 대표할 수 있는 정보라고 할 수는 없다.

결과적으로, 제안한 라우팅 알고리즘은 자신의 잔류전원, 네트워크상의 traffic 부하, 이동(mobility), hop count를 이용한 거리 등을 종합적으로 평가한 Cost함수를 이용함으로써 기존의 라우팅 알고리즘과 비교하여 좀 더 안정적이고 지속적인 라우팅 경로를 택할 수 있음을 보였다. 또한 실험의 복잡도를 줄이기 위해 실험에서는 보이지 않았지만, 제안한 알고리즘이 이동성을 가진 센서 노드로서의 멀티에이전트 로봇시스템이 라우팅 경로에 참여할 수 있는 확률을 떨어뜨림으로써 네트워크 전체의 효율을 떨어뜨릴 수 있는 link broken을 줄일 수 있다는 것은 쉽게 유추할 수 있다. 이것은, 역으로 멀티에이전트 로봇시스템이 임무를 수행함에 있어 보다 안정적인 네트워크 환경을 가지는 것을 의미한다.

또한 라우팅 비용(Cost) 함수의 경우, 네트워크의 크기나 목적에 따라 파라메타를 임의로 추가할 수도 있고 각 파라메타의 weight를 적절하게 조절하고 선택할 수 있기 때문에 상황에 맞는 좀 더 유연한 방법을 제공한다. 즉 전원의 소모가 많다고 할지라도 배터리의 교체가 용이한 경우 식 (1)의 w_{11} 는 작게 설정할 수 있음을 의미한다.

(a) Route discovery by hop.

(b) First link broken by N5 power-off and route recovery by hop.

(c) Second link broken by N7 power-off and route recovery by hop.

(d) Third link broken by N6 power-off and route recovery by hop.

그림 20. Hopping count에 의한 route discovery (con't).

Fig. 20. Route discovery by hopping count (con't).

(a) Route discovery by cost function.

(b) First link broken by N11 power-off and route recovery by cost function.

(c) Second link broken by N7 power-off and route recovery by cost function.

(d) Third link broken by N5 power-off.

그림 21. Cost function에 의한 route discovery (con't).

Fig. 21. Route discovery by Cost function (con't).

그림 22. 각 알고리즘에 따른 전체 노드들의 power 추이도.

Fig. 22. Power tendency of every node on each algorithm.

표 2. Link broken 발생 시점 비교.

Table 2. Comparison of the link broken time.

algorithm	1	2	3
Hop	2240sec	2540sec	3120sec
Cost function	2240sec	2930sec	3260sec

향후 과제로는 경로가 결정된 이후에 센서 노드의 추가 등으로 인하여 더 좋은 경로가 생긴 경우에 자동으로 이를 감지하고 경로를 업데이트하는 방법에 대한 추가적인 연구가 필요하겠다.

참고문헌

- [1] T. Finin, R. Fritzson, and R. McEntire, "KQML as an agent communication language," *Proc. of the 3rd International Conference on Information and Knowledge Management*, Nov. 1994.
- [2] Foundation for intelligent physical agents. (1997) [Online]. Available: <http://www.fipa.org>
- [3] K. Sycara, K. Decker, A. Pannu, M. Williamson, and D. Zeng, "Distributed intelligent agents," *IEEE Expert*, Dec. 1996.
- [4] H. Chalupsky et al., "Electric elves: Agent technology for supporting human organizations," *AI Magazine*, Summer 2002.
- [5] K. Decker, K. Sycara, and D. Zeng, "Designing a multi-agent portfolio management system," *Proc. of the AAAI Workshop on Internet Information Systems*, 1996.
- [6] O. Shehory, G. Sukthankar, and K. Sycara, "Agent aided aircraft maintenance," *Proc. of Autonomous Agents '99*, pp. 306-312, May 1999.
- [7] M. Berna-Koes, I. Nourbakhsh, and K. Sycara, "Communication Efficiency in multi-agent systems," *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, pp. 2129-2134, 2004.
- [8] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, pp. 102-114, Aug. 2002.
- [9] S. Madden, M. J. Franklin, and J. M. Hellerstein, "The design of an acquisitional query processor for sensor networks," *Proc. ACM SIGMOD*, ACM Press, pp. 491-502, 2003.
- [10] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TAG: A Tiny AGgregation service for AD-Hoc sensor networks," *Proc. Symp. OSDI*, 2002.
- [11] M. Demirbas and H. Ferhatosmanoglu, "Peer-to-peer spatial queries in sensor networks," *3rd International Conference on Peer-to-Peer Computing (P2P 2003)*, pp. 32-39, Sep. 2003.
- [12] Y. Zou and K. Chakrabarty, "Coverage -oriented sensor deployment," *Distributed Sensor Networks*, R. R. Brooks and S. S. Iyengar (eds.), CRC Press, pp. 453-481, 2005.
- [13] S. S. Iyengar, A. Tandon, Q. Wu, E. Cho, N. S. V. Rao, and V. K. Vaishnavi, "Deployment of sensors : An overview," *R. R. Brooks and S. S. Iyengar (eds.), CRC Press*, pp. 483-504, 2005.
- [14] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," *Comp. Commun. Rev.*, pp. 234-244, Oct. 1994.
- [15] C. E. Perkins and E. M. Royer, "Ad Hoc on-demand distance vector routing," *Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, pp. 90-100, Feb. 1999.
- [16] D. B. Johnson, "Routing in Ad Hoc networks of mobile hosts," *Proc. of the Workshop on Mobile Computing Systems and Applications*, pp. 158-163, Dec. 1994.
- [17] D. B. Johnson and D. A. Maltz, "Dynamic source routing in Ad Hoc wireless networks," *Mobile Computing*, edited by Tomasz Imielinski and Hank Korth, Kluwer Academic Publishers, pp. 153-181, 1996.

김 준 업

2012년 중앙대학교 전자전기공학부(공학사). 2012년~현재 중앙대학교 대학원 전자전기공학부 석사 1차. 관심분야는 Brain-Computer Interface System, Particle Swarm Optimization, Neuro-Robotics 등.

박 승 민

2010년 중앙대학교 전자전기공학부(공학사). 2010년~현재 중앙대학교 대학원 전자전기공학부 석·박사 통합과정. 관심분야는 Brain-Computer Interface, Intention Recognition, 소프트웨어 컴퓨팅 등.

고 광 은

2007년 중앙대학교 전자전기공학부(공학사). 2007년~현재 중앙대학교 대학원 전자전기공학부 석·박사 통합과정 수료. 관심분야는 Human-Computer Interaction, Brain-Computer Interface System, Intention Recognition, Neuro-Science,

Neuro-Robotics, 소프트웨어컴퓨팅 등.

장 인 훈

1993년 중앙대학교 제어계측공학과 졸업(공학사). 1999년 중앙대학교 대학원 제어계측공학과(공학석사). 2010년 중앙대학교 대학원 전자전기공학부(공학박사). 1999년~2000년 (주)신도리코 기술연구소 연구원. 2000년~2009년 (주)

아로마솔루션 기술연구소 책임연구원. 2009년~현재 한국생산기술연구원 선임연구원. 2011년~현재 과학기술연합대학원대학교(UST) 지능형로봇공학과 교수. 관심분야는 Machine Learning, Multi-Agent Robotic Systems (MARS), Probabilistic Robotics, Rehabilitation Robot 등.

심 귀 보

1984년 중앙대학교 전자공학과(공학사). 1986년 중앙대학교 전자공학과(공학석사). 1990년 The University of Tokyo 전자공학과(공학박사). 1991년~현재 중앙대학교 전자전기공학부 교수. 2002년~현재 중앙대학교 중소기업산업

협력센터 센터장. 2006년~2007년 한국지능시스템학회 회장. 2007년~현재 (사)한국산학연합회 서울지역협회 회장. 2009년~2010년 중앙대학교 중앙도서관장 및 박물관장. 2011년~현재 중앙대학교 스마트지능로봇연구센터 센터장. 관심분야는 인공지능, 뇌-컴퓨터 인터페이스, 의도인식, 감성인식, 유비쿼터스 지능형로봇, 지능시스템, 컴퓨테이션 인텔리전스, 지능형 홈 및 홈 네트워크, 유비쿼터스 컴퓨팅 및 센서 네트워크, 소프트 컴퓨팅(신경망, 퍼지, 진화연산), 다개체 및 자율분산로봇시스템, 인공지능시스템, 지능형 감시시스템 등.