



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

석사학위논문

RFID를 이용한 실내 위치추적 방법에 관한연구

Research on Position Tracking Using Passive RFID Tags

제 출 자 : 이 창 환

지도교수 : 조 재 형

2011

산업공학과

산업공학전공

단국대학교 대학원

RFID를 이용한 실내 위치추적 방법에 관한연구

Research on Position Tracking Using Passive RFID Tags

이 논문을 석사학위논문으로 제출함.

2011년 12월

단국대학교 대학원
산업공학과
산업공학전공

이 창 환

이창환의 석사학위 논문을
합격으로 판정함

심 사 일 : 2011. 12. 13.

심사위원장 김 봉 진 인

심 사 위 원 장 경 인

심 사 위 원 조 재 형 인

단국대학교 대학원

(국문초록)

RFID를 이용한 실내 위치추적 방법에 관한연구

단국대학교 대학원 산업공학과

산업공학전공

이 창 환

지도교수 : 조 재 형

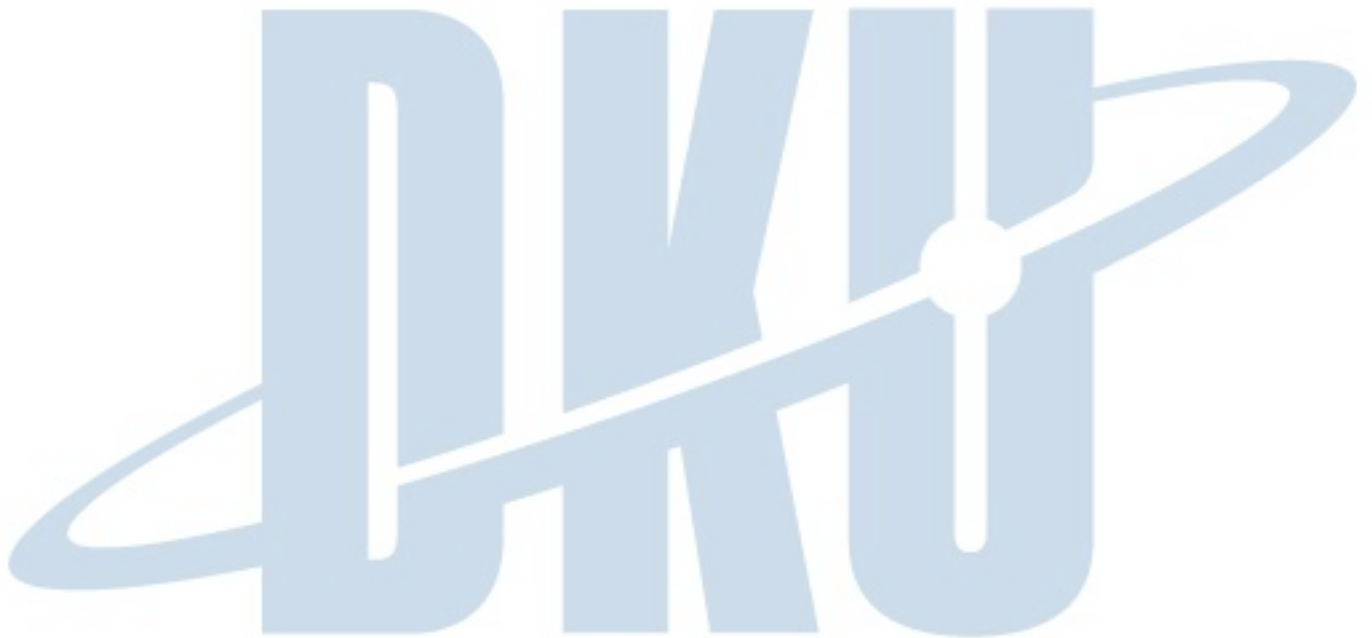
위치추적시스템은 이동로봇, 인터넷의 위치기반서비스(LBS) 등 다양한 분야에서 응용이 가능하다. 이러한 위치추적 시스템 중에서 현재 가장 알려진 시스템은 GPS 시스템이나, 실외에서 사용될 수 있도록 위성을 통해서 위치를 추적하는 시스템이며, 실내에서는 GPS 수신 장치가 제대로 작동되지 않아 문제가 발생한다.

실내 위치추적 시스템으로는 Wi-Fi 위치시스템과 블루투스의 사용 환경 기술에 의존하고 있으며, 이 장치들은 1~10m 정도의 정밀도를 보이고 있다. 이 시스템으로 위치를 추적하는 것은 시간과 투자, 비용 면에서 실효를 거두고 있지 못한 상황이다. 실내 LBS 시스템을 위해서 RFID 시스템의 도입이 요구되어왔으며 비용효과와 정밀도를 보장하는 실내에서의 위치추정 장치의 개발이 필요한 시점이다.

그러나 대부분의 RFID를 이용한 시스템은 능동 인식 RFID 태그를 사용한 방법으로 배터리를 이용한 능동형 태그를 사용하여야 한다. 수동형 RFID 태그를 이용한 실내의 위치를 추적하는 시스템에서는 정밀도와 정확도에서 문제점을 안고 있다.

본 논문에서는 실시간 저비용 실내 위치추정 시스템을 RFID 시스템을 이용하여 설계, 개발 하였으며, 실내에서의 보다 정밀한 위치 추적을 할 수 있도록 다양한 위치추적 알고리즘을 적용하여 결과를 비교하였다.

본 연구의 결과로 실내에서 작동하는 RFID 태그를 이용한 저가의 위치 추적 장치를 설계 개발하였다. 개발된 시스템은 반영구적 수동형 RFID 태그를 이용한 시스템이며, Wi-Fi를 이용한 시스템보다 저가이며 상대적으로 높은 정밀도를 보장할 수 있다. 또한, 실내 위치추적 시스템을 실험에 의하여 타당성을 검증하였으며, 시그널의 오작동이나 잡음에 의해 생긴 오차를 보정하는 방법을 제시하였다. 룩업테이블에 의한 보정 대신에 B-spline 곡면식을 보간에 의해 생성한 후 기준 데이터로 사용하였으며, 개발된 시스템은 이동로봇, AGV, 스마트폰 위치기반 서비스(LBS) 등에 활용할 수 있다.



목 차

I. 서 론	1
1. 연구목적	1
2. 연구방법과 논문구성	2
II. 이론적 고찰	4
1. RFID의 특성	4
2. RFID 시스템의 활용 현황	8
3. 수동형 RFID 태그를 이용한 위치추적 시스템	9
4. GPS를 이용한 실내 위치추적 시스템	12
III. 본론	14
1. 태그 인식 데이터 분석	15
3. 패턴매칭에 의한 위치추적	19
4. 태그패턴에 의한 중심좌표 보정 방법	22
5. B-spline 곡면식에 의한 보정 위치추적	24
IV. 결과 및 토의	27
V. 결 론	41
참고문헌	43
부 록	44
영문요약	55

표 목차

<표 2-1> RFID의 기본 구성요소	5
<표 2-2> RFID 태그 분류	6
<표 2-3> RFID와 다른 유사매체와 비교	7
<표 2-4> RFID주파수의 종류	7
<표 2-5> Discrete-Simplified Sensor Model 근사화	11
<표 3-1> 태그인식 확률데이터	15
<표 3-2> 각 노드의 오차분포	17
<표 4-1> 각 좌표의 중심 값	29
<표 4-2> 각 좌표의 중심 값의 오차	29
<표 4-3> 실험데이터	30
<표 4-4> 중심 값에 의한 실험 데이터	32
<표 4-5> 중심 값에 의한 각도	33
<표 4-6> 2진 연산에 의한 매칭 비율	33
<표 4-7> 2진 연산에 의한 실험 데이터	36
<표 4-8> 2진 연산에 의한 각도	36
<표 4-9> <표 4-1>에서 찾은 실험데이터 위치	37
<표 4-10> 비례식을 이용한 실험데이터	38
<표 4-11> 비례식을 이용한 각도	38
<표 4-12> 근사곡면으로 하는 제어점	39
<표 4-13> B-spline 곡면식을 이용한 실험데이터	39
<표 4-14> B-spline 곡면식을 이용한 각도	40

그림목차

<그림 2-1> 인식률.....	10
<그림 2-2> 샘플 포인트	10
<그림 2-3> 공통영역.....	11
<그림 2-4> 센서 공간에서의 태그좌표.....	12
<그림 2-5> 외부 단말기와 협력을 이용한 실내 측위 기법.....	13
<그림 3-1> 태그인식 거리와 리더기 인식거리	14
<그림 3-2> 태그인식 확률데이터 그래프.....	16
<그림 3-3> 각 노드의 오차분포 그래프.....	17
<그림 3-4> 인식된 태그의 중심좌표.....	18
<그림 3-5> 각 노드의 2진코드와 셀 2진코드	20
<그림 3-6> 태그 번호에 의해서 생성된 2진코드	20
<그림 3-7> 패턴매칭 비율을 결정하기 위한 순서도.....	21
<그림 3-8> 각 노드의 실험좌표값과 평균좌표.....	22
<그림 3-9> 위치 추적 시스템의 구축 방법.....	26
<그림 4-1> IU-9060 RFID 리더기와 수동형 태그.....	27
<그림 4-2> 좌표입력 화면.....	28
<그림 4-3> 태그 측정 화면.....	28

I. 서 론

1. 연구목적

위치추적시스템은 육지측량, 비행항공술, 이동로봇, 가상현실 등 많은 분야에서 중요하게 사용되고 있다. 스마트폰이나 인터넷의 위치기반서비스(LBS)의 위치추적시스템은 넓은 지역을 인식할 수 있어야 하며, 핸드폰이나 컴퓨터와 함께 장치나 태그의 형태로 사용되어야 한다. 이러한 시스템은 많은 분야에서 응용이 가능하며, 응급상황, 자동차 내비게이션 시스템, 여행자 투어 계획, 광고 맵 등 정보 공유차원에서 사용될 수 있다.

무선 추적 시스템으로 GPS, 무선전화 추적 시스템, 와이파이 위치 시스템, RFID 추적 시스템 등 많은 위치 추적 장치들이 개발되고 있다. 이와 같은 장치들은 각각 용도에 맞는 장단점을 갖고 있다. 이러한 위치추적 시스템 중에서 현재 가장 알려진 시스템은 GPS 시스템이다. 일반적으로 GPS 시스템으로부터 정보를 받아서 현재 위치에서 주변의 정보나 데이터를 장치에 다운 받아 사용하게 된다. 이 장치는 실외에서 사용될 수 있도록 위성을 통해서 위치를 추적하는 시스템이며, GPS 시스템은 위성 수신 상태에 따라 모든 장소의 위치추정에는 한계가 있으며, 특히 실내에서는 GPS 수신 장치가 제대로 작동되지 않아 문제가 발생한다. GPS 신호는 대부분의 건축구조물에 차단되므로 실내에서는 무용지물이며, 제약적으로 사용되고 있다.

일반적인 실내 위치추적 시스템은 실외에서 실행되는 위치추적 시스템과는 약간 다른 방법을 사용되고 있다. 실내의 추적시스템은 현존하는 Wi-Fi 위치시스템과 블루투스의 사용 환경 기술에 의존하고 있으며, 이 장치들은 1~10m 정도의 정밀도를 보이고 있다. Wi-Fi 네트워크는 통신수단으로서 사용되고 있으며, 이 시스템으로 위치를 추적하는 것은 시간과 투자, 비용 면에서 실효를 거두고 있지 못한 상황이다. 정밀도를 높이기 위한 Wi-Fi AP 시스템은 실내 환경에서 상당히 복잡하고 보통의 경우, 결과가 정밀도 면에서 좋지 못한 결과를 가져온다. 실내 환경에서 와이파이 시스템의 부정확성뿐만 아니라, 비용 측면에서도 높은 비용을 초래한다. 실제로 실내 위치추적 시스템은 비용에 비례하는 설치비용이 필요하고, 공간에 대한 지역적인 제한이 요구된다. 실내 LBS 시스템을 위해서 RFID 시스템의 도입이 요구되어왔으며 90년대 말부터 연구가 시작되면서 새로운 시장의 새로운 연구가 시도 되었다. Wi-Fi 시스템에 의한 위치추적이 가능하더라도 높은 비용의 Wi-Fi 태그가 사용되기 때문에 가격 측면에서 고려해야 할 사항이다. Wi-Fi 시스템에 의한 위치추적 장치가 있더라도 넓은 범위의 장소에서 사용하기 위해서는 많은 와이파이 태그를 사용해야 하므로 비싼 개발비와 비용을 초래하게 된다. 그러므로 비용효과와 정밀도를 보장하는 실내에서의 위치추정 장치의 개발이 필요한 시점이다.

현재 여러 가지 실내 위치추적시스템과 무선 네트워크 장치로서 실내 RFID 위치 추적 시스템에 대한 연구가 활발하게 진행되고 있다. RFID 기술은 비접촉식 자동 인식 기술이며, 이는 라디오 신호로 인식, 추적 배열 감지 등의 역할을 할 수 있으며, 사람, 자동차, 상품 등에 신호를 감지하여 사용할 수 있다. RFID 기술은 물체의 움직임을 네트워크나 라디오 스캔 장치에 의해서 몇 미터에서 수십 미터까지의 거리에서 인식한다. 이러한 장점에 의해서 많은 연구자들이 RFID를 이용한 위치추적 시스템을 연구하고 있다. Fu, A. & Retscher, G.,(2009)는 RFID 태그의 위치를 계산하였으며, Zhou, Y. et al. (2007)은 새로운 인공 로케이션 시스템을 자동 로봇에 적용하여 사용하였다. 이러한 시스템은 실내에서 작동하는 GPS 시스템과 같은 역할을 하였다. Mor, T. et al. (2008)은 지문 방식 시스템으로 타겟의 위치를 판별하였다.

그러나 이 모든 시스템은 능동 인식 RFID 태그를 사용한 포지셔닝 방법으로 이러한 시스템은 모두 배터리를 이용한 능동형 태그를 사용하여야 한다. 수동형 RFID 태그를 이용한 실내의 위치를 추적하는 시스템에서는 정밀도와 정확도에서 문제점을 안고 있다.

본 논문에서는 실시간 저비용 실내 RFID 위치추정 시스템을 이용한 내용을 다루며, 실내에서의 보다 정밀한 위치 추정을 할 수 있도록 다양한 위치추적 알고리즘을 적용하여 결과를 비교한다. 이 시스템은 AGV, 자동로봇 위치 추적 장치, 이동 장치의 위치 추적 장치 등에 적용할 수 있으며, 수동형 RFID 태그를 사용하여 저비용의 시스템을 구축 할 수 있다.

2. 연구방법과 논문구성

수동형 RFID 태그를 이용한 물체의 위치추적 시스템은 움직이는 물체를 RFID 리더장치를 장착하고 바닥 혹은 천정에 고정되어 있는 태그를 이용하여 위치를 추적한다. 위치 추적의 방법으로는 여러 위치에서 인식하는 태그들의 데이터를 수집하여 감지되는 태그들의 패턴과 데이터를 분석하여 현재의 위치를 인식하는 여러 알고리즘을 개발하여 적용한다.

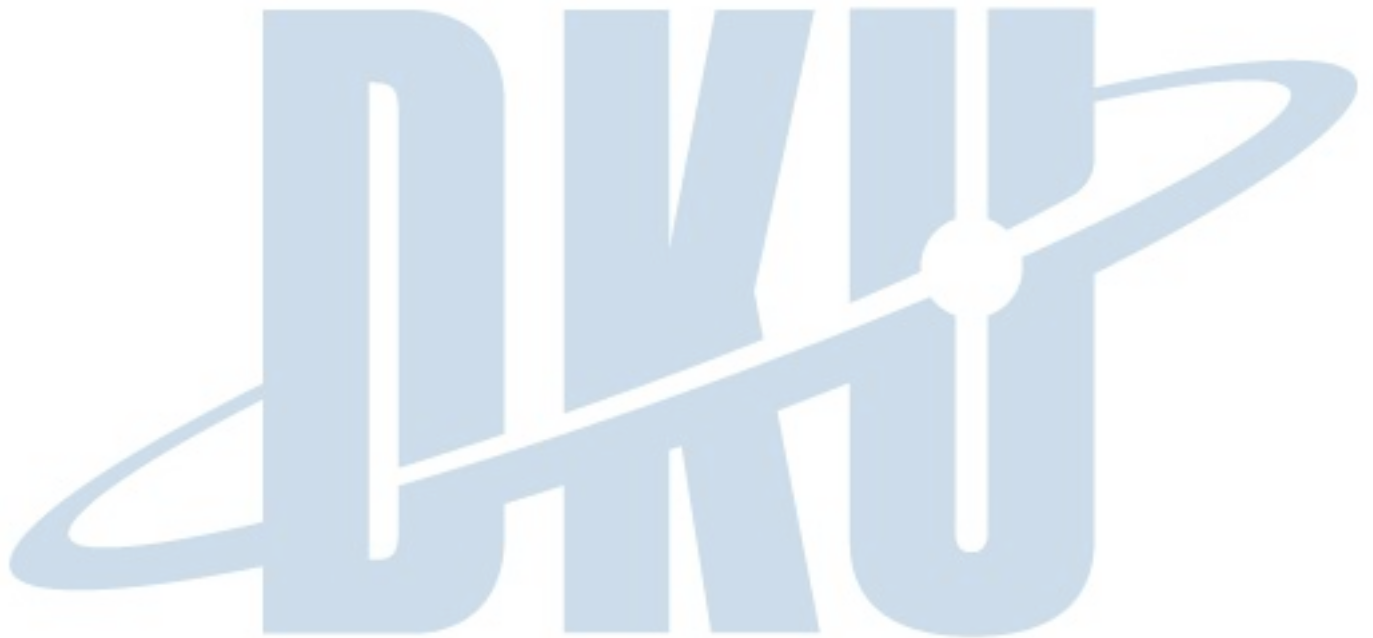
일반적인 방법으로 태그 위치의 면적 중심점을 계산하여 위치를 인식하는 방법과 비교되는 세 가지의 위치 추적 방법을 적용하여 움직이는 물체의 정보를 추적, 분석, 개선하는 방법에 대하여 결과를 비교한다.

태그들의 특성은 위치마다, 환경에 따라서 약간씩 다른 양상을 보이므로 데이터 분석을 통하여 여러 가지 가능한 위치 추적 알고리즘을 개발한다. 감지한 시그널을 분석하여 패턴매칭과, 면적중심 계산, 통계적 가중치를 준 중심점, 룩업테이블, 곡면방정식 피팅 등의 방법에 의해서 개발된 여러 가지 추적 방법을 통하여 데이터를 비교한다.

여러 가지 추적 알고리즘을 비교하여 분석하며, 정밀도를 높일 수 있는 방법을 제시하며, 실험을 통하여 제시된 추적 방법의 우수함을 증명한다.

논문의 구성은 서론에서 연구의 목적을 정의하고 2장에서는 실내외에서 실행되는 여러 가지

위치추적 방법에 대하여 고찰하였으며, 특히 RFID를 이용한 여러 가지 위치추적 방법을 알아보았다. 3장에서는 위치를 추적하는 방법을 제시하여 이론을 정리하였다. 중심점을 구하는 방법과, 패턴매칭 방법을 제시하였으며, 면적중심 룩업테이블에 의한 위치보정 방법과 곡면피팅에 의한 위치보정 방법에 의한 위치추적 방법을 제시하였다. 4장에서는 연구의 결과를 검증하기 위하여 제시된 위치추적 방법에 대하여 실험을 실시하여 제시된 알고리즘의 성능을 검증하였다. 마지막으로 결론에서 연구의 요약 및 한계점과 향후 과제를 제시하였다.



II. 이론적 고찰

자동인식(AIDC) 기술의 한 종류인 RFID(Radio Frequency Identification)는 마이크로칩을 내장한 태그에 저장된 데이터를 무선주파수를 이용하여 비접촉으로 읽는 기술로 태그 정보를 무선으로 수 미터에서 수 십미터까지 보내며 리더는 태그의 신호를 받아 정보를 컴퓨터로 보내는 기술이다. 그러므로 관리할 대상 사물에 태그를 부착하고 전파를 이용하여 사물 및 주변 상황정보를 감지하고 필요한 정보를 수집, 저장, 가공, 축적함으로써 사물에 대한 측위, 원격처리, 관리 사물간 정보 교환 등 다양한 서비스를 제공하는 시스템을 말한다.





RFID 리더에는 태그를 향하여 전파를 주고받는 전자회로 부분을 가지고 있으며 리더 내에 마이크로프로세서는 태그로부터 들어오는 신호를 바꿔주거나 그 데이터의 신호를 검증하면서 기억장치인 메모리에 저장하기도 하며 필요에 따라서는 나중에 송신하기도 한다. 안테나는 전파를 주고받을 수 있는 전자회로부분과 같이 케이스에 포함되어 있거나 단독으로 분리되어 있는 경우도 있다. 태그에는 IC Chip과 연결된 안테나와 전파동조를 위한 콘덴서가 내장되어 있다. 전자태그의 태그는 크게 배터리 내장여부와 주파수 대역에 따른 구분으로 나누어질 수 있다.

1. RFID의 특성

RFID의 기본 구성요소는 태그, 안테나, 리더, 호스트로 나누어져 있으며 각 요소 별 원리는 <표 2-1>과 같다.

RFID의 태그에 대해 살펴보면 RFID 태그는 전원공급의 유무에 따라 전원을 필요로 하는 능동형(Active)과 내부나 외부로부터 직접적인 전원의 공급없이 리더기의 전자기장에 의해 작동되는 수동형(Passive)형으로 구분된다. 능동형 타입은 리더기의 필요전력을 줄이고 리더와의 인식거리를 멀리할 수 있다는 장점이 있으나, 전원 공급장치를 필요로 하기 때문에 작동시간의 제한을 받으며 수동형에 비해 고가인 단점이 있다. 반면, 수동형은 능동형에 비해 매우 가볍고 가격도 저렴하면서 반영구적으로 약 10년 이상 사용이 가능하지만, 인식거리가 10m 이내이며, 리더기에서 더 많은 전력을 소모한다는 단점이 있다. 또, RFID 태그는 기능에 따라 Read-Only, WORM(Write Once Read Many), Read/Write의 3가지로 분류할 수 있다. Read Only 태그는 제조 시 기록되며 정보 내용의 변경이 불가능하다. 그러나 가격이 저렴하여 단순인식을 요하는 RFID분야에 사용된다. WORM태그는 사용자가 데이터를 프로그램하며 프로그램한 후에는 변경이 불가능하다. Read/Write 태그는 몇 번이고 프로그램 및 데이터 변경이 가능한 구조이다.

<표 2-1> RFID의 기본 구성요소

구성요소	원 리
<div>태 그</div> 	<ul style="list-style-type: none"> - 상품에 부착되며 데이터가 입력되는 IC 칩과 안테나로 구성 - 리더와 교신하여 데이터를 무선으로 리더에 전송 - 배터리 내장 유무에 따라 능동형과 수동형으로 구분됨
<div>안 테 나</div> 	
<div>리 더 기</div> 	<ul style="list-style-type: none"> - 주파수 발신을 제어하고 태그로부터 수신된 데이터를 해독함 - 용도에 따라 고정형, 이동형, 휴대용으로 구분 - 안테나 및 RF회로, 변/복조기, 실시간 신호처리 모듈, 프로토콜 프로세서 등으로 구성
<div>호스트</div> 	<ul style="list-style-type: none"> - 한개 또는 다수의 태그로부터 읽어 들인 데이터를 처리함 - 분산되어 있는 다수의 리더 시스템을 관리함 - 리더부터 발생하는 대량의 태그 데이터를 처리하기 위해 에이전트 기반의 분산 계층 구조로 되어 있음

<표 2-2> RFID 태그 분류

방식별 구분		원 리
읽기/쓰기 가능여부	읽기전용	<ul style="list-style-type: none"> - 제조 시 정보 입력, 정보내용은 변경 불가 - 가격이 저렴하여 바코드와 같이 단순인식 분야 사용
	한번쓰기 가능	<ul style="list-style-type: none"> - 사용자가 데이터를 1회 입력할 수 있으며 입력 후에는 변경 불가
	읽기/쓰기 가능	<ul style="list-style-type: none"> - 여러 번 데이터 입력과 변경이 가능 - 가격은 높지만 고가 상품 등에 활용 가능
태그 전원유무	능동형 (Active)	<ul style="list-style-type: none"> - 태그에 배터리가 부착, 수십m 원거리 통신용 - 가격 고가, 수명 제한, UHF대역이상에서사용
	수동형 (Passive)	<ul style="list-style-type: none"> - 태그에 배터리가 없으며, 10m 이내 근거리 통신용 - 가격 저렴, 수명 반영구적(약 10년이상)

RFID와 다른 유사매체와 비교해 보면 그 특징을 확연히 알 수 있다. 다른 유사 인식 매체별 인식 기술을 비교해보면 인식방법에서는 RFID는 비접촉식으로 바코드에 비해 인식속도가 빠른 특징을 가지고 있다. 또한, 바코드의 인식거리는 최대 50cm 인데 반해, RFID는 최대 100m 까지 확장이 가능하며, 금속을 제외한 장애물의 투과도 가능하다. 인식률에 있어서도 자기카드나 IC카드와 마찬가지로 99.9% 이상으로 높으며, 사용기간 및 데이터 저장 능력 또한 여타 매체에 비해서 탁월하다. 다만, RFID 태그의 가격이 타 인식 매체에 비해 고가이기 때문에 빠른 실용화를 위해서는 가격의 인하가 동반되어야 한다.

<표 2-3> RFID와 다른 유사매체와 비교

	바코드	자기카드	IC 카드	RFID
인식방법	비접촉식	접촉식		비접촉식
인식거리	~50cm	직접 삽입		~100m
인식속도	4초	4초	1초	0.01~0.1초
인식률	95% 이하	99.9% 이상		
데이터 저장	1~100byte	1~100byte	16~64Kbyte	64Kbyte 이하
데이터 쓰기	불가	가능		
재활용	불가		가능	

RFID주파수의 종류는 저주파수 대역, 중간주파수, 대역 고주파수 대역, 마이크로파 대역으로 나누어지며 주파수별로 인식 거리가 달라 적용 분야에 차이를 보인다.

<표 2-4> RFID주파수의 종류

주파수 구분	저주파수 대역 (125kHz &134kHz)	중간주파수 대역 (13.56MHz)	고주파수 대역 (433MHz)	고주파수 대역 (860~960MHz)	마이크로파 대역 (2.45GHz)
인식 거리	60cm미만	60cm까지	50~100m	3.5~10m	1m이내
특 징	- 저가형 - 느린 인식속도	- 중저가형 - 상호유도방식 적용 - 비금속 장애물의 투과성 우수	- 고가형 - 능동형 - 긴 인식거리	- 저가형 - 금속 및 액체 인식을 저조 - 수동형	- 빠른 인식속도 - 차폐물이 있는 경우 인식 불가 - 고가형
적용 가능분야	- 출입통제 - 동물식별 - 재고관리	- 출입 통제 - 스마트카드	- 컨테이너 식별 및 추적	-유통물류 분야	-자동차 운행 흐름 모니터링 - 톨게이트 시스템

2. RFID 시스템의 활용 현황

ISO 11784, 14223은 134kHz 대역의 동물관리용 RFID의 규격을 규정하고 있다. 동물식별에서는 무선주파수에 의한 동물식별 코드구성 및 동물식별 기술지침이 규격화되어 있다. 동물관리용 식별코드는 모두 64BIT로 구성된다. 동물관리용 RFID는 광우병 등과 같이 돌연히 병이 발병한 가축의 이력파악이나 은폐 방지, 농장의 자동화에 매우 효과적이다. 또한 길 잃은 애완동물 수색에도 위력을 발휘한다. 최근에는 동물의 건강관리나 번식에도 활용하고 있다.

ISO 10374는 구체적으로 RFID 태그를 판독하는 데이터 시퀀스로서 소유자 코드나 컨테이너 형식, 길이, 높이, 폭, 적재, 중량, 공적중량 등을 규정하고 있다. 해상에서의 악천후에 견딜 수 있도록 완전방수나 내열, 내충돌, 진동구조, 전자환경 속에서도 동작하는 등의 물리적 특성도 배려한 규격이다. 통신특성으로서는 최대 13m의 통신 영역 내를 태그가 130km/h로 통과할 경우에도 올바른 판독이 가능해야 한다. 등록번호나 화물주 등이 기입된 해상 컨테이너용 RFID 태그를 해상운송용 컨테이너에 부착함으로써 항만이나 육상 수송중인 컨테이너의 위치나 적재물의 내용 등에 관한 정보를 실시간으로 취득할 수 있고, 수송효율의 향상에 도움을 줄 수 있다.

비접촉형 IC카드의 표준화 작업은 1985년 워싱턴회의에서 표준화에 대한 대응정책이 결정되었다. 당시는 접점형 IC카드의 문제점으로서 접점의 접촉 불량 등을 해결하여 정전기에 의한 IC의 파괴방지, 카드의 삽입방향에 대한 자유도 확보, 비접촉형인터페이스 가능 등의 요구를 실현할 목적으로 비접촉화의 검토가 시작되었다. 최초로 접점을 전자결합으로 교체하는 '밀착형'에 대하여 심의하였다. 그 후 1994년 4월의 고베회의에서 더욱 편리한 사용을 위해서 전송거리를 늘릴 수 있는 비접촉형 카드의 표준화 제안을 채택하였다. 심의가 진행됨에 따라 비접촉형 카드는 제품에 대한 기술적 차이, 특히 통신거리와 차이에 따라 밀착형, 근접형, 근방형의 3종류로 분리되었다. ISO/IEC 14443은 한국의 버스카드, NTT의 텔레폰카드, 오오사카의 CITY카드 등이 있다.

ISO/IEC JTC/SC31 WG4의 심의대상은 SCM과 관련된 물품관리용 RFID로서, 선행한 ISO/IEC JTC1/SC17(IC카드)이나 ISO TC204(고도 교통 시스템) 등과 보완 관계로 협조적으로 심의를 추진하고 있다. 현재 WG4는 응용을 조사하는 ARP 그룹이나 전파법제도를 조사하는 레귤러 트리그룹 이외에 데이터 구문을 심의하는 SG1, 고유식 별자를 심의하는 SG2, 주파수나 변조도 등의 에어 인터페이스를 심의하는 SG3이 설치되어 12항목의 IS 또는 TR(기술 리포트)를 담당하고 있다. 물품관리용 RFID 태그는 ID와 정보저장 영역을 보유하는 다목적용관 ID만 가지는 저가의 단일 기능용으로 구분할 수 있다[8].

RFID를 물품에 부착함으로써 우선 화물의 출발 지점, 경유 지점, 도착 지점 등 화물의 물리적 이동에 대한 실시간 관리가 가능해진다. 예를 들면 공항의 수화물 취급소에서 아무리 기다

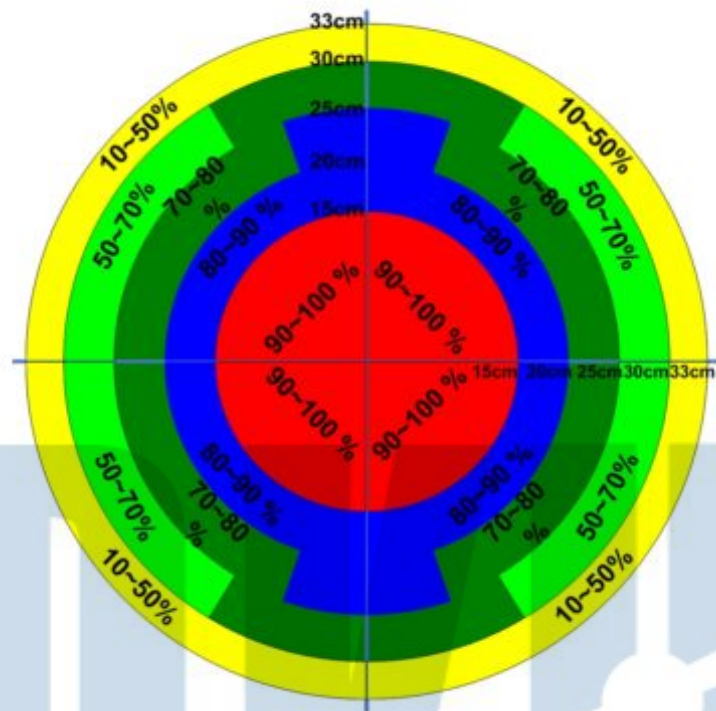
려도 출발할 때 맡겼던 옷 가방이 도착하지 않아 애를 태웠던 경험을 가지고 있는 사람이 많을 것이다. 이러한 사실은 현재 수화물에 붙어있는 바코드나 태그가 도움이 되지 않는다는 증거로 볼 수 있으며, 바코드나 태그를 RFID로 교환함으로써 공하의 수화물 분류가 확실하면서도 신속하게 처리되어 오배송을 방지한다.

또한, RFID를 사용하여 물품의 위치 관리가 불필요하게 한다. 예를 들어 책에 RFID를 부착해두면 이용자가 도서관 사서에게 책 제목만 가르쳐주어도, 책이 소장되어있는 장소를 파악할 수 있기 때문에 즉석에서 해당 서적을 찾아볼 수 있게 된다. 또 항만이나 공항 컨테이너 기지에서 검색 대상 컨테이너 및 물품 위치에 대해 바로 인식할 수 있기 때문에 컨테이너 위치 관리가 불필요하다.

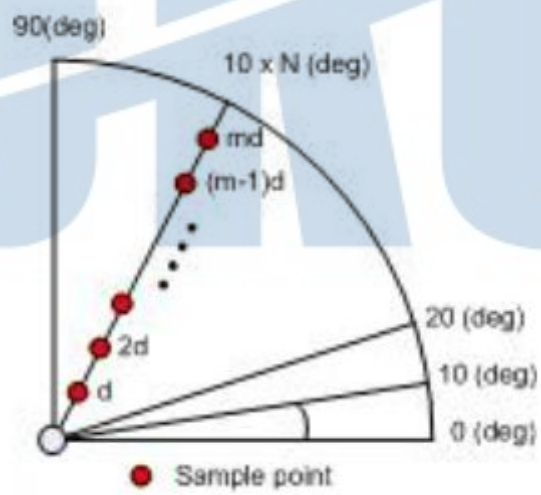
사무 처리의 효율성이 향상 될 수 있다. 예를 들면 슈퍼마켓에서는 상품 카트에 상품을 실은 채로 정산 게이트를 통과하기만 하면, 즉석에서 정산이 완료된다. 이들 기능을 빌딩이나 가옥의 모든 문과 창에 응용하는 일은 비교적 용이하다. 또 지폐, 수표, 주식과 티켓 등에 RFID를 심어놓으면 위조나 부정 유통을 쉽게 방지할 수 있다[7].

3. 수동형 RFID 태그를 이용한 위치추적 시스템

지용관(2009) 등은 RFID Tag를 기반으로 한 이동 로봇의 경로 계획을 세우기 위하여 13.56MHz의 주파수 영역대의 RFID를 사용하였다. 13.56MHz의 주파수를 이용한 RFID 안테나의 최대 인식거리는 30cm이다. 4.2m x 7.5m 공간에 10cm 간격으로 태그를 배치하였다. 이동 중 한 개의 태그를 인식하였을 경우와 다수의 태그를 인식하였을 경우 2가지의 추정기법을 이용하여 이동로봇의 위치와 방위를 추정하였다. 먼저 한 개의 태그만 인식 하였을 경우에는 확률 센서 모델을 이용하여 적은 연산량으로 로봇에 상대적인 위치와 방위를 추정할 수 있다. 좌표계는 이동 로봇이 태그가 인식되는 시작 위치를 원점으로 하는 좌표계이며 매 측정순간마다 인식의 방향성과 인식률을 이용하여 확률 센서 모델로부터 태그와의 상대거리와 상대방향을 추정하게 된다. 이 논문에서 사용한 확률 센서 모델의 RFID의 인식률은 그림1과 같으며, Discrete-Simplified Sensor Model은 실험결과를 10(deg) 단위로 샘플 포인트 (d) 간격으로 나누어 인식확률을 구하였다.<그림 2-1>, <그림 2-2>, <표 2-5>[2]



<그림 2-1> 인식률

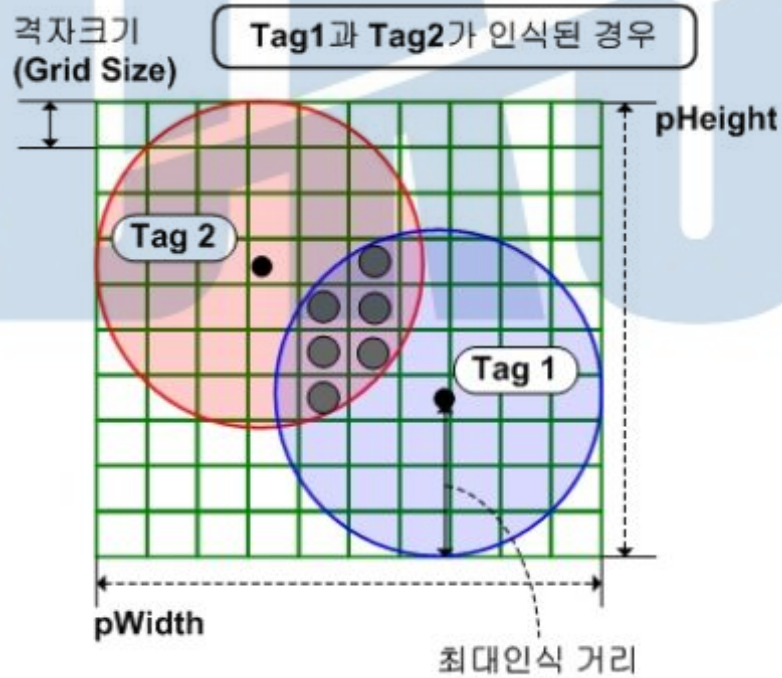


<그림2-2> 샘플 포인트

<표 2-5> Discrete-Simplified Sensor Model 근사화

Deg dist	0	10	20	30	40	50	60	70	80	90
d	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95
2d	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95
3d	0.95	0.95	0.95	0.95	0.85	0.85	0.95	0.95	0.95	0.95
4d	0.85	0.85	0.85	0.85	0.75	0.75	0.75	0.82	0.85	0.85
5d	0.75	0.75	0.75	0.6	0.6	0.6	0.6	0.6	0.85	0.85
6d	0.6	0.6	0.3	0.3	0.3	0.3	0.3	0.3	0.75	0.75
7d	0	0	0	0	0	0	0	0	0	0

이동 로봇이 이동 중에 다수의 RFID 태그를 인식하였을 경우에 사용되는 기법은 태그간의 공통영역을 도출한다. 공통영역은 최대 인식 거리를 이용하여 인식된 태그와 태그 사이에서 생성된다. 공통영역을 도출하기 위하여 격자 영역의 크기를 결정하고, 최대 인식 거리에 따른 원 방정식을 사용하여 공통영역을 도출한다.<그림2-3>[2]

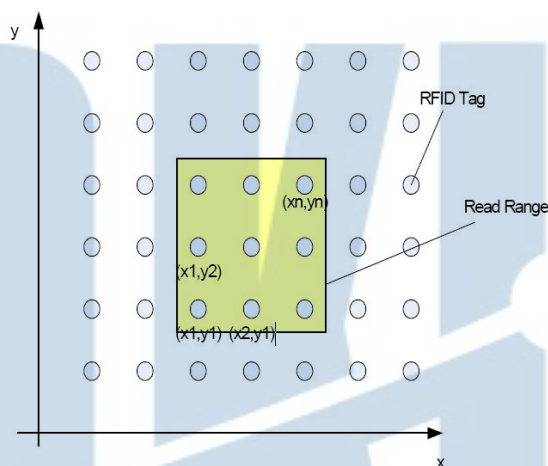


<그림 2-3> 공통영역

김원(2007)은 리더기가 태그를 인식하였을 경우 로봇의 중심 좌표 (x_{est}, y_{est}) 는 식 (1)과 같이 계산하였다.

$$\begin{aligned} x_{est} &= \frac{\max(x_1, \dots, x_n) + \min(x_1, \dots, x_n)}{2} \dots\dots\dots (1) \\ y_{est} &= \frac{\max(y_1, \dots, y_n) + \min(y_1, \dots, y_n)}{2} \end{aligned}$$

(단, n은 리더기에 감지된 태그의 수)로 정의 하였다.[5]

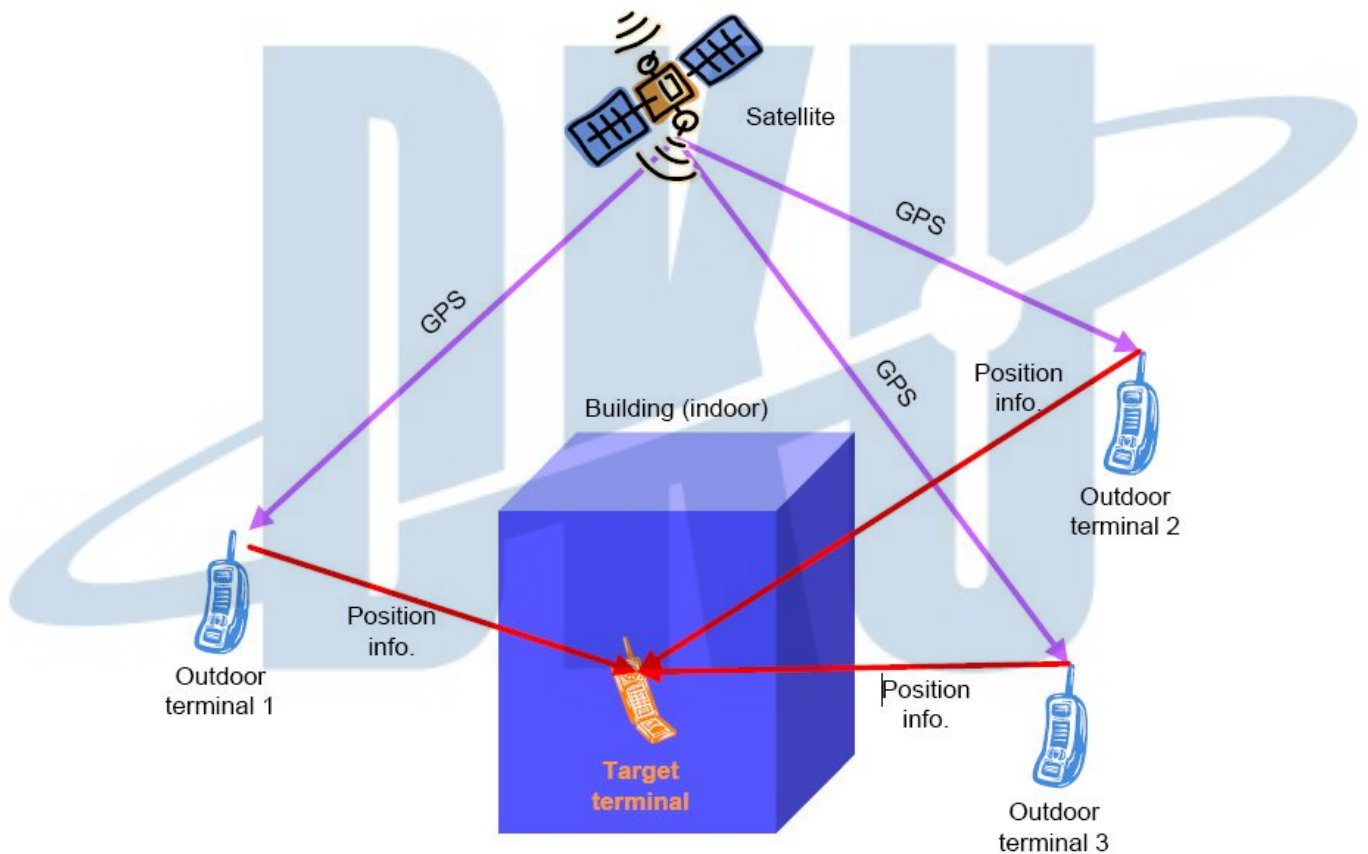


<그림 2-4> 센서 공간에서의 태그좌표

RFID 태그에 의해서 인식되는 위치 추적 시스템은 읽혀진 태그 데이터의 좌표를 이용하여 각 축에서의 최대, 최소값을 평균 내거나 모든 좌표의 평균에 의해서 중심좌표를 구하는 방법을 사용하게 된다. 그러나 이런 방법에 의한 위치 추적 방법은 모서리 부분에서 많은 오차를 포함하게 되며, 전파가 안정적으로 작동하지 않고 주변 환경에 많은 영향을 받고 잡음이 발생하기 때문에 계산된 좌표는 많은 오차를 포함한 값이다. 그러므로 같은 환경에서 정밀도를 향상하기 위해서 오차를 제거할 수 있는 다른 방법이 연구되어야 한다.

4. GPS를 이용한 실내 위치추적 시스템

조형민(2010) 등은 GPS를 이용한 실내 위치추적을 하기 위하여 외부 모바일 단말기를 이용하였다. GPS가 탑재된 외부 모바일 단말기를 이용하여 실내에 존재하는 단말기의 위치 추적하는 모델은 <그림 2-5>와 같이 설명 하였다. GPS를 이용하면 실외에 위치하고 있는 단말기의 위치를 비교적 정확하게 측정할 수 있어 이러한 단말기들을 이용하여 자신의 위치를 포함한 신호를 전송하게 되면 실내에 위치한 단말기는 외부 단말기에서 전송된 신호의 도착한 시간을 통한 시간차를 이용하여 신호의 속도를 고려하여 거리를 계산 위치를 추적하는 기법이다.[9]

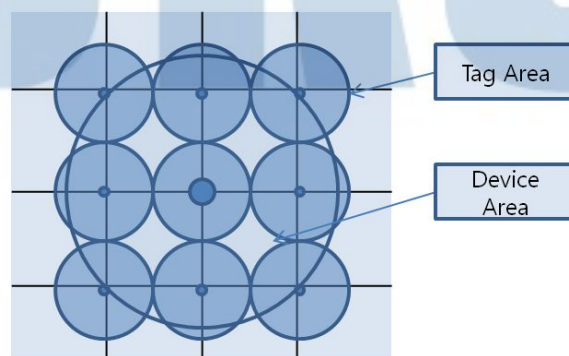


<그림 2-5> 외부 단말기와 협력을 이용한 실내 측위 기법

이 방법의 장점으로서는 보급되어 있는 무선 단말기를 이용하기 때문에 별도의 물리적 장치가 필요 없다고 말하고 있다. 하지만 주위에 GPS가 탑재된 외부 모바일 단말기가 없거나 유동인구가 적은 곳에서는 측정에 어려움이 있다.

III. 본론

본 연구의 목적은 RFID 수동형 태그를 이용한 이동 장치의 위치를 추적하는 방법을 개발하는 것이다. 위치를 추적하는 방법으로 일반적으로 리더기의 위치에서 감지된 RFID 태그 시그널의 위치를 파악하여 그 데이터를 이용하게 된다. 기록된 태그가 하나라면 그 태그의 위치가 현재 리더기의 위치가 된다. 이 경우 태그 한 개의 위치에 의해서 위치가 계산되므로 정밀도 면에서 태그의 간격과 비례하여 많은 오차를 포함하게 된다. 정밀도를 보다 높이기 위해서 리더기의 범위 내에 많은 태그가 감지 될 수 있도록 두는 것이 바람직하다. 이 경우 태그들의 위치들을 계산하여 태그가 포함하고 있는 면적 중심을 리더기의 위치로 쉽게 계산하여 파악 할 수 있다. 위치를 추적하기 위한 태그의 배치 문제는 여러 가지 방법이 있을 수 있으나, 태그의 위치를 효율적으로 배치하기 위하여 일정한 간격을 갖는 격자 형태로 태그를 배열하여, 태그의 위치를 노드로 하고 사방에서 감싸는 태그의 면적을 셀로 하여 리더기의 위치를 감지되는 노드의 위치에 의하여 리더기의 위치 좌표를 구할 수 있다. 이상적으로 리더기의 감지 범위를 원으로 두고 그 범위 안에 있는 태그 위치의 면적 중심점을 구하면 <그림 3-1>과 같이 오차 범위 내에서 리더기의 위치를 구할 수 있다. 그러나 리더기와 태그가 갖고 있는 안테나 특성과 주위의 환경과 습도, 온도에 따라서 다양하게 신호의 세기가 변화하기 때문에 이론적으로 항상 일정한 신호와 범위 영역을 단정 짓기에는 어려움이 있다. 또한 모서리 부분에서는 태그 평균값에 의해서 중심이 공간의 안쪽으로 이동함에 따라 커다란 오차를 포함하게 된다. 이렇게 오차와 환경에 쉽게 변하는 시스템의 특성을 고려하는 여러 가지 위치 추적 방법을 제시하고자 한다.



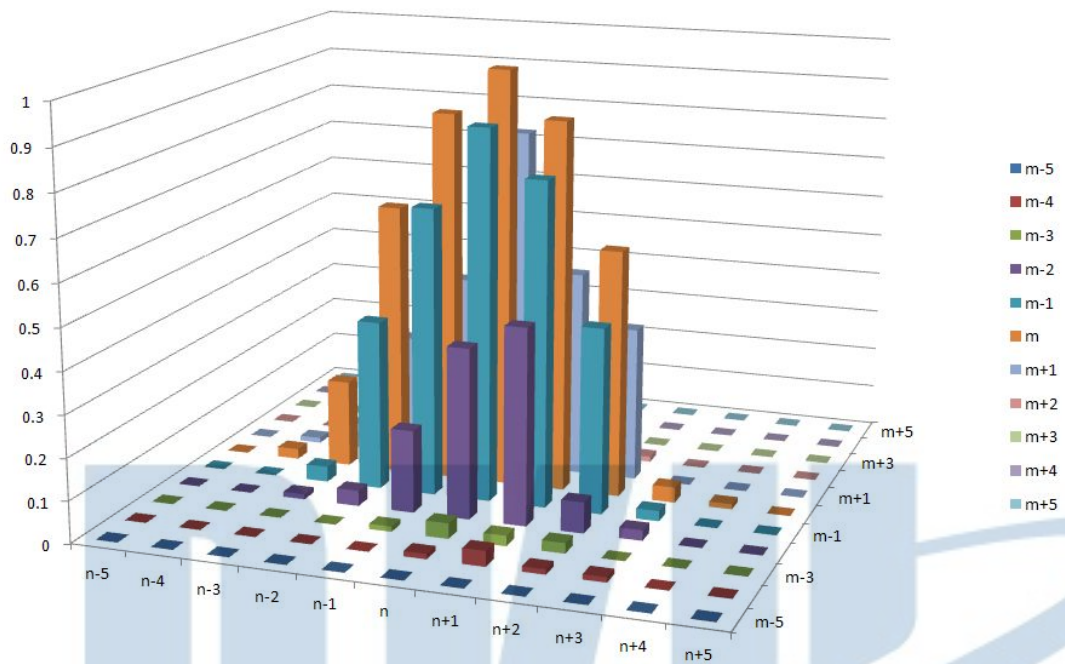
<그림 3-1> 태그인식 거리와 리더기 인식거리

1. 태그 인식 데이터 분석

리더기에서 감지하는 태그 데이터를 분석하면, 이상적인 환경에서 항상 일정한 태그 인식률을 보이면서 리더기 중심으로 거리가 멀어짐에 따라 고르게 인식확률이 분포되어야 하지만, 실제 데이터는 리더기 위치에 따라서, 인식되는 태그에 따라서 그 인식확률이 수시로 변화함을 알 수 있다. 다음 <표 3-1>는 리더기 중심에서 확인한 태그의 인식 확률 데이터이다. 태그가 정확하게 원을 그리면서 거리에 따른 인식확률이 분포되는 것이 아님을 알 수 있다. <그림 3-2>에서 보듯이 태그 인식률이 일정하지 않으며, 특히 리더기에서 멀어질수록 예측할 수 없는 확률을 보이는 것을 알 수 있다.

<표 3-1> 태그인식 확률데이터

	n-5	n-4	n-3	n-2	n-1	n	n+1	n+2	n+3	n+4	n+5
m-5	0	0	0	0	0	0	0	0	0	0	0
m-4	0	0	0	0	0	0.012	0.037	0.012	0.012	0	0
m-3	0	0	0	0	0.012	0.037	0.024	0.024	0	0	0
m-2	0	0	0.012	0.037	0.197	0.407	0.469	0.074	0.024	0	0
m-1	0	0	0.037	0.407	0.691	0.888	0.777	0.444	0.024	0	0
m	0	0.024	0.209	0.654	0.888	1	0.888	0.592	0.037	0.012	0
m+1	0	0.012	0.061	0.296	0.456	0.827	0.493	0.370	0	0	0
m+2	0	0	0.012	0.049	0.111	0.135	0.061	0.012	0	0	0
m+3	0	0	0	0	0.012	0.024	0.012	0	0	0	0
m+4	0	0	0	0	0	0	0	0	0	0	0
m+5	0	0	0	0	0	0	0	0	0	0	0

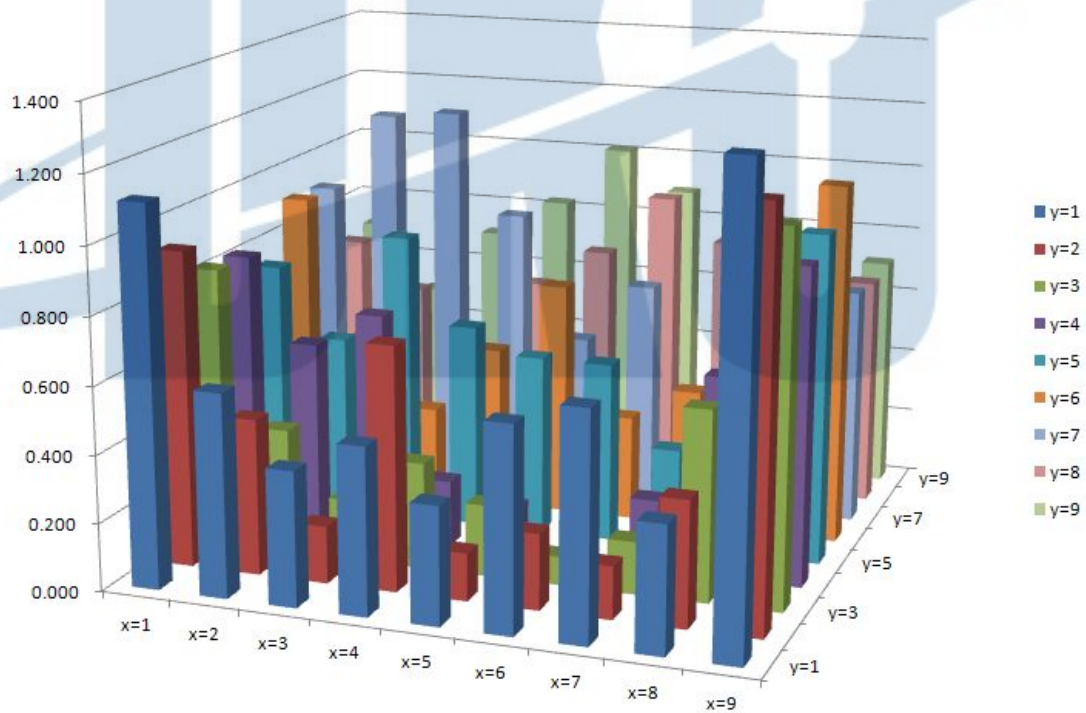


<그림 3-2> 태그인식 확률데이터 그래프

<표 3-2>는 리더기를 모든 영역에서 작동하여 태그를 인식한 후, 태그 좌표들의 평균을 구한 결과이다. 전 지역에서 오차를 포함하고 있으며, 특히 모서리 지역에서 많은 오차를 보이고 있다. 모서리 지역에서의 오차는 평균을 취하면서 중심좌표가 안쪽으로 위치하는 방향으로 잘못 계산되어진 결과이며, 내측의 오차는 인식률이 거리에 비례하여 고르게 분포하지 않은 결과이다. <그림 3-3>은 오차의 분포를 그래프로 보여준다.

<표 3-2> 각 노드의 오차분포

	x=1	x=2	x=3	x=4	x=5	x=6	x=7	x=8	x=9
y=1	1.118	0.599	0.398	0.492	0.348	0.600	0.665	0.371	1.360
y=2	0.935	0.462	0.170	0.720	0.141	0.224	0.157	0.371	1.204
y=3	0.834	0.370	0.180	0.316	0.212	0.085	0.156	0.563	1.093
y=4	0.825	0.574	0.682	0.196	0.141	0.296	0.210	0.597	0.936
y=5	0.747	0.537	0.870	0.613	0.538	0.539	0.297	0.406	0.979
y=6	0.915	0.489	0.279	0.489	0.707	0.316	0.417	0.667	1.077
y=7	0.910	1.156	1.177	0.867	0.489	0.674	0.337	0.679	0.710
y=8	0.691	0.550	0.361	0.602	0.721	0.910	0.782	0.270	0.691
y=9	0.707	0.500	0.707	0.825	1.012	0.890	0.873	0.500	0.707

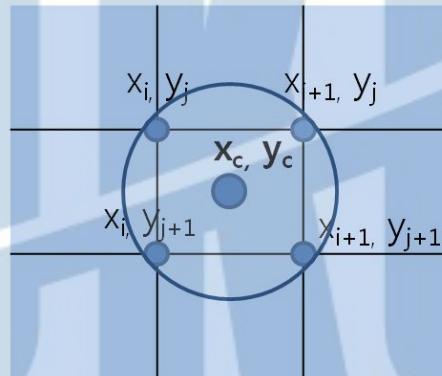


<그림 3-3> 각 노드의 오차분포 그래프

2. 태그 면적중심 계산법에 따른 위치추적

리더기가 태그를 감지하는 면적을 항상 일정하고 정확하게 계산하기는 불가능하다. 신호의 세기가 방사각 별로 일정하지 않고 변화하며 태그 상호간의 반응에 의해서도 리더기 반응이 다르게 나타나기도 한다. 그렇지만 태그들의 위치는 일정한 면적의 반응영역을 갖고 있다고 가정하여 리더기에 반응한 태그들의 위치좌표를 이용하여 중심을 구한 좌표가 리더기 위치를 나타내는 좌표의 확률이 높아진다. 다음 <그림 3-4>와 같이 각 태그의 위치 데이터의 면적중심 값이 리더기의 좌표로 추적하는 방법이다. 2진화된 좌표의 면적 계산은 다음과 같다.

$$A = \sum_{i=1}^{N_c} \sum_{j=1}^{N_r} b_{ij} \dots\dots\dots (2)$$



<그림 3-4> 인식된 태그의 중심좌표

여기서 b_{ij} 는 0 또는 1의 값을 갖는 2진화 변수 값이며, N_r, N_c 는 정수로 나누어진 행과 열의 수이다. 중심좌표 (x_c, y_c) 는 다음과 같이 구할 수 있다.

$$x_c = \frac{m_{10}}{m_{00}}, y_c = \frac{m_{01}}{m_{00}} \dots\dots\dots (3)$$

여기서 m_{pq} 는 물체의 x, y 좌표에 따른 값으로 다음의 식에 의해서 계산된다.

$$m_{pq} = \sum_{i=1}^{N_c} \sum_{j=1}^{N_r} b_{ij} (x_i)^p (y_j)^q \dots\dots\dots (4)$$

계산 결과에 의한 중심좌표 (x_c, y_c) 가 리더기의 중심좌표이며, 일반적으로 가정하는 이동로봇의 위치 좌표이다. <그림 3-4>에서 보는 바와 같이 중앙에 커다란 원의 영역에 리더기 인식 범위에 놓이게 되면 원 내부의 태그에 대하여 데이터를 읽어 인식하게 된다. 중앙에 인식영역이 줄어들거나 늘어나게 되어도 영역 안에 들어 있는 태그의 위치들의 중심 좌표를 구하면 그 위치를 리더기 중심 위치로 인식하게 된다.

그러나 중심좌표 (x_c, y_c) 를 추정좌표로 사용하기에는 많은 문제를 포함하고 있다. 모서리 부분에서 중심좌표는 태그좌표들의 중심좌표를 계산하는 것이므로 모서리에서 멀어지는 값으로 중심좌표가 얻어진다. 또한 신호에 잡음이 들어가게 되면 오차가 큰 값을 포함하게 되어 단순한 중심좌표 계산만으로는 구한 좌표의 신뢰도가 떨어지게 되는 문제점을 안고 있다. 보다 신뢰도를 높이는 방법을 보장하기 위하여 이러한 여러 가지 문제점을 보완할 수 있는 방법이 제시되어야 한다.

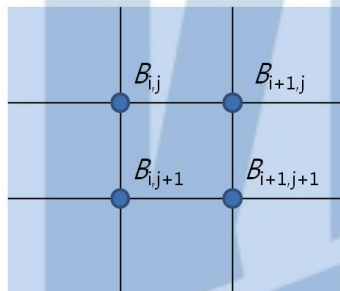
3. 패턴매칭에 의한 위치추적

중심좌표에 의한 위치 추적 방법에는 모든 태그의 무게점을 같이 두어 계산하였다. 그러나 실제로 태그를 감지하는 횟수가 각 위치에 따라 많이 감지되기도 하고, 현격하게 낮은 비율로 감지되는 경우가 있다. 또한 중심좌표에서부터 멀어지는 거리만큼 확률이 줄어드는 것이 아니라, 실제 실험 데이터에 따르면 각 리더기의 위치에 따라 반응하는 태그가 각각 다르게 나타나는 경향이 있다. 따라서 각 위치에서 태그 감지 확률이 리더기 위치에 따라 다른 비율로 분포하게 된다. 그러나 이러한 확률은 안테나 특성과 태그의 안테나 특성에 따라 정확한 이론값을 부여하는 데는 문제가 있다. 이 값은 실험값에 의해서 계산하는 것이 더욱 합리적 방법이다. 상대적 분포를 알아보기 위해서 모든 위치에서 리더기 중심과 거리에 따른 감지 확률을 알아볼 수 있다. 각 노드에서 읽은 실험값에 의한 확률 변수에 의해서 가중치를 갖게 하여 면적 중심좌표를 계산하는 것이 더욱 위치 반응 신뢰도를 높일 수 있다.

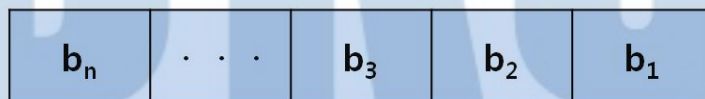
리더기 위치추적은 태그 중심좌표를 계산하는 방법 외에 리더기의 각각 위치에서 반응하는 태그 그룹의 인식에 의해서도 위치를 추적할 수 있다. 리더기의 위치에 따라 반응하는 태그 그

룹의 패턴이 각각 다르게 나타난다. 반응하는 태그 그룹들은 어느 정도의 잡음을 포함하고 있으나, 대체적인 패턴과 리더기에 반응하는 태그 그룹들은 크게 변화하지 않으므로 위치에 따른 태그 그룹들의 특성을 이용하여 좌표를 추적할 수 있다.

각 노드에서 감지되는 태그들은 각각 서로 다른 형태로 나타난다. <그림 3-5>과 같이 그리드의 노드에 태그를 장치하고 각 노드에서 리더기를 사용하여 태그 데이터를 읽었을 때 나타나는 2진화된 노드 값들이다. $B_{i,j}$ 는 (i,j) 노드에서 인식한 태그데이터를 2진 데이터로 하여 숫자로 나타낸 것이다. 이 2진 데이터는 <그림 3-6>와 같이 각 노드를 자리수로 하는 이진 코드이다. 각 비트는 태그 아이디의 감지 여부를 나타낸다. n 개의 태그를 각 비트의 자리수로 하고 감지된 비트는 숫자 1로, 감지되지 않은 태그는 0으로 하여 이진 숫자를 생성한다.



<그림 3-5> 각 노드의 2진 코드와 셀 2진 코드

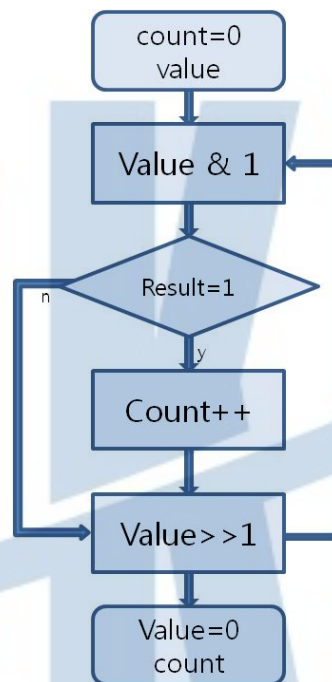


<그림 3-6> 태그 번호에 의해서 생성된 2진코드

생성된 2진 숫자는 노드의 속성 데이터로 사용한다. 리더기에서 인식한 태그 데이터는 중심 좌표를 구하지 않고 노드 속성 데이터 $B_{i,j}$ 와 연산을 하기 위하여 인식한 태그들을 이용하여 2진 코드 R 을 생성한다. 2진수 R 은 인식된 태그들을 자리수로 하는 코드이며, 임계값을 설정하여 수준 이하의 숫자가 인식된 태그들은 삭제하며, 분리되어 떨어져 있는 태그가 인식되었을 때도 잡음으로 간주하여 삭제 시킨다. 생성된 2진 코드는 잡음을 제거하여 정밀도를 높인 데이터이다. R 과 각 노드의 2진 코드와 AND 연산을 수행하여 결과 값을 취한다.

$$Pb_{i,j} = \frac{(1's\text{갯수})(R \cdot C_{i,j})}{(1's\text{갯수})(R)} \dots\dots\dots (5)$$

비트 연산값은 매칭비율 $Pb_{i,j}$ 이다. 여기서 bC 는 이진 숫자에서 0이 아닌 1의 개수를 나타내는 연산자이며, ‘.’은 and 비트 연산자이다. 이진 숫자에서 0이 아닌 1의 개수를 세는 방법의 순서도는 <그림 3-7>와 같다.



<그림 3-7> 패턴매칭 비율을 결정하기 위한 순서도

리더기 위치 추적 방법은 각 노드에서 얻어진 매칭비율 $Pb_{i,j}$ 값을 이용하여 태그의 중심좌표를 구하는데, 각 노드를 똑같은 무게값으로 구하는 대신에 각 노드에 가중치를 두어 중심좌표를 계산한다. 가중치는 매칭비율 $Pb_{i,j}$ 로 한다. 이때 태그 인식 과정에서 발생할 수 있는 잡음을 제거하기 위하여 임계값을 두어 임계값 이하의 매칭비율은 제거하고 계산한다. 중심좌표 $(xp_{i,j}, yp_{i,j})$ 를 구하기 위한 수식은 다음과 같다.

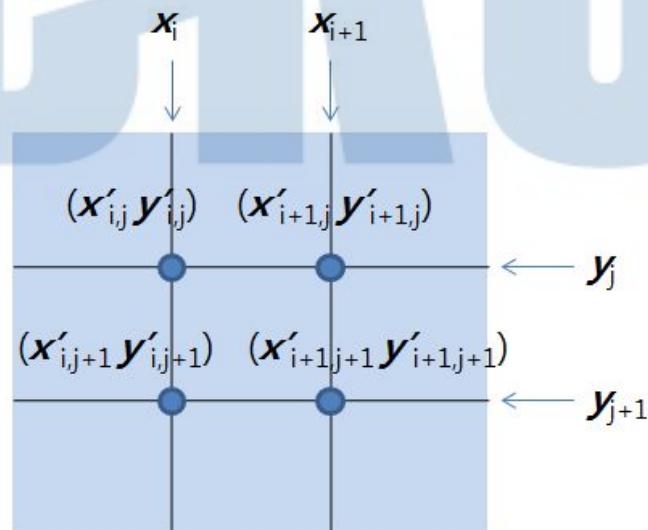
$$xp_{i,j} = \frac{m_{10}}{m_{00}}, \quad yp_{i,j} = \frac{m_{01}}{m_{00}} \dots\dots\dots (6)$$

$$m_{pq} = \sum_{i=1}^{N_c} \sum_{j=1}^{N_r} Pb_{ij} (x_i)^p (y_j)^q \dots\dots\dots (7)$$

태그 패턴에 의해 구한 리더기 위치 추적 방법은 리더기에 인식된 태그를 같은 가중치로 두지 않고 확률에 따른 변수로 두어 계산하여 신뢰도를 향상할 수 있다. 그러나 모서리 부근에서는 위치에 대한 잡음을 완전히 제거할 수 없으므로 오차를 포함하게 된다. 이 문제점을 해결하기 위해서 실험에 의한 오차를 고려하여 제거하는 방법이 요구된다.

4. 태그패턴에 의한 중심좌표 보정 방법

모서리에서 오차가 큰 문제는 각 노드와 셀에서 오차의 방향과 크기를 계산하여 발생되는 오차를 보정하여 해결 할 수 있다. 오차의 방향과 크기는 각 노드에서의 방향벡터에 의해서 결정된다. 각 노드의 오차 방향과 크기는 실험값에 의해서 구하며, 이 보정 값에 의해서 모서리에서 발생하는 오차를 보정하여 해결한다. <그림 3-8>은 각 노드에서의 오차 방향과 크기의 예이다.



<그림 3-8> 각 노드의 실험좌표값과 평균좌표

예를 들면, $(x'_{i,j}, y'_{i,j})$ 는 각각 좌표 i,j 노드에서의 리더기에 의해서 얻어진 태그들의 중심좌표이다. 같은 방법에서 모든 노드에서 태그들의 정보를 인식하여 중심좌표를 노드 데이터 (x', y') 를 LUT(Look-up Table)로 저장한다. 이동로봇이나 위치를 추적하기 위한 리더기의 위치에서 RFID 태그의 위치를 인식한 데이터와 각 노드 데이터 값을 비교하여 리더기 장치의 위치를 평균좌표에 의한 비례연산으로 구한다.

<그림 3-8>에서 계산에 의한 중심좌표가 4개 노드의 사이 셀에 위치하는 좌표일 경우 x좌표는 $x'_{i,j}$ 와 $x'_{i+1,j}$ y좌표는 $y'_{i,j}$ 와 $y'_{i,j+1}$ 사이 값이 된다. 4개의 노드에 둘러싸인 셀의 위치를 구하기 위해서 다음과 같은 평면 비례식을 이용하여 해를 구할 수 있다.

$$xl = x_i + \frac{2x_c - x'_{i,j} - x'_{i,j+1}}{x'_{i+1,j} + x'_{i+1,j+1} - x'_{i,j} - x'_{i,j+1}} \dots\dots\dots (8)$$

$$yl = y_j + \frac{2y_c - y'_{i,j} - y'_{i,j+1}}{y'_{i+1,j} + y'_{i+1,j+1} - y'_{i,j} - y'_{i,j+1}} \dots\dots\dots (9)$$

(xl, yl) 는 평면 비례식을 이용한 x, y의 좌표이며, (x_c, y_c) 는 리더기에 의해서 얻어진 태그좌표의 중심이다. 이 경우 모서리에서 발생할 수 있는 오차와 잡음에 의해 발생하는 데이터 오류를 현저하게 제거하여 물체의 위치 좌표를 결정할 수 있다.

이 방법은 중심좌표를 구한 후, 이미 측정한 각 노드 값에 의한 오차를 선형으로 데이터에 저장하여 x, y축만 확인하여 비례식에 의해서 빠르게 위치를 추적할 수 있다는 장점이 있다. 그러나 잡음이 들어간 신호를 완전히 제거할 수 없기 때문에 생각하지 못한 오차를 포함할 수 있으며, 미리 만들어 놓은 룩업테이블의 데이터의 오류를 감지하여 개선하는 시스템이 없으므로 환경 변화에 대한 잡음의 오차에 대해서 신뢰도를 보장할 수 없다.

5. B-spline 곡면식에 의한 보정 위치추적

록업테이블에 의한 위치 추적 시스템은 모서리 부분에서의 오류를 보정할 수 있는 장점이 있다. 록업테이블을 사용하는 대신에 곡면 방정식을 사용하여 신속하게 계산 결과에 의해서 위치를 추적하는 방법을 제시한다. 록업테이블에 의한 위치추적 방법은 데이터를 비교하여 각 셀에서의 위치를 직선 비례식에 의해서 다시 계산하여야 한다는 번거로움이 있다. 이 점을 개선하기 위하여 B-spline 곡면식을 이용하여 데이터를 곡면으로 보간하여 록업테이블 없이 사용할 수 있게 하였다. B-spline 곡면식은 다음과 같이 쓸 수 있다.

$$Q(u,v) = \sum_{i=0}^m \sum_{j=0}^n B_{i,j} S_{i,j}(u,v) \dots\dots\dots (10)$$

여기서 $B_{i,j}$ 는 조정점을 나타내며, $S_{i,j}$ 는 매개변수 u, v 에 대한 Basis 함수이다. 이 곡면식은 행렬식으로 다음과 같이 쓸 수 있다.

$$[Q] = [S][B] \dots\dots\dots (11)$$

여기서 $[Q]$, $[S]$, $[B]$ 는 각각 곡면의 데이터좌표, 베이스함수, 제어점좌표들을 나타낸다. 이 곡면식을 주어진 데이터 좌표에 의해서 얻으면 행렬 $[Q]$ 를 $[D]$ 로 교체하여 쓸 수 있다.

$$[D] = [S][B] \dots\dots\dots (12)$$

여기서 $[D]$ 는 리더기에 의해서 얻어진 태그 좌표들의 중심좌표이다.

태그 중심좌표들에 의해서 제어점 좌표 $[B]$ 를 구하기 위해서는 다음과 같은 식에 의해서 구해야 한다.

$$[B] = [S]^{-1}[D] \dots\dots\dots (13)$$

$m \times n$ 사각 폴리곤 네트에 의해서 만들어진 데이터라면 행렬 $[D]$ 는 삼차원 좌표계에서 $(m \times n) \times 3$ 의 변수식으로 구해진다. 데이터 포인트가 많아질수록 동시에 만족해야할 좌표수가 늘어나므로 해를 구하는 일은 최적화 알고리즘을 적용해서 풀어야 할 문제가 된다. 효과적으로 해를 구하는 방법으로 Wang et al(1993)이 제안한 방법이 있다. 구하고자 하는 제어점은 곡면상의 모든 주어진 점을 지나는 곡면 보간방법이 된다. 이때 제어점은 곡면상의 데이터 좌표수와

일치하게 된다. 주어진 데이터 좌표에 의해서 보간데이터를 구하려면 각각의 노드에 매개변수 u, v 값이 주어져야하며, u, v 값은 범위는 다음과 같이 주어진다.

$$0 \leq u \leq 1, 0 \leq v \leq 1 \dots\dots\dots (14)$$

따라서 각 노드의 실제 좌표를 각각 수직 수평선의 길이를 나누어서 범위 0에서 1까지가 되도록 정규화하여 매개변수 u, v 값을 분리하여 곡면식에서 해를 구한다. 따라서 B-spline 곡면은 매개변수와 실제 좌표의 증가율이 다르게 나타나는 non-uniformed 곡면을 생성하게 된다. 곡면보간을 위해서 제어점 $B_{i,j}$ 를 구하여야 하며, 다음 식을 만족하여야 한다.

$$p_{i,j}^{k+1} = p_{i,j}^k + (D_{i,j} - \sum_{i=0}^m \sum_{j=0}^n B_{i,j} \cdot S_{i,j}(u,v)) \dots\dots\dots (15)$$

여기서 초기값 $p_{i,j}^0$ 은 측정데이터 $D_{i,j}$ 로 주어지며 $(D_{i,j} - Q_{i,j})$ 는 $p_{i,j}^k$ 의 제어계수로 사용하는데 k번째 해의 오차량으로 간주하여 해를 구하는 방법이다. 반복수행은 에러값이 주어진 공차 내에 들어올 때 까지 계속된다. 여기서 목적함수 $F(u,v)$ 는 다음과 같다.

$$F(u,v) = (x_m - x(u,v))^2 + (y_m - y(u,v))^2 \dots\dots\dots (16)$$

여기서 (x_m, y_m) 과 $(x(u,v), y(u,v))$ 는 만들어진 곡면 위의 포인트와 수식에 의한 곡면 좌표이다.

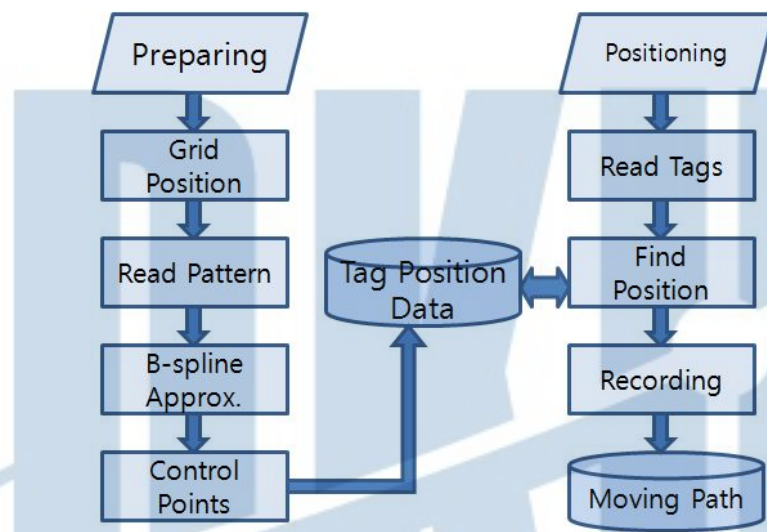
록업테이블을 이용하는것과 다른 점은 B-spline 수식을 이용하여 포지셔닝을 단순화 할 수 있으며, 태그 중심좌표가 계산되면 록업테이블을 찾아 비례식을 계산하는 대신에 식(17)과 같이 리더기의 실제 위치를 바로 계산에 의해서 찾을 수 있다.

$$(x_c, y_c) \rightarrow g(u,v) \rightarrow h(x'_c, y'_c) \dots\dots\dots (17)$$

여기서 좌표 (x_c, y_c) 는 매개변수식 $g(u,v)$ 에서 평면상의 가장 근접거리를 만족하는 매개변수 (u,v) 를 구하고, 실제 좌표와 환산하여 (x'_c, y'_c) 를 얻을 수 있다. 따라서 (u,v) 는 태그 좌표에 의한 매개변수 값이며, (x'_c, y'_c) 는 결정 좌표이다.

본 논문에서 제시하는 위치 추적 시스템의 구축 방법은 다음 <그림 3-9>과 같이 정밀도의 요

구사항에 맞춰서 그리드를 생성하여 태그를 부착한다. 전 지역에 걸쳐서 리더기를 이용하여 태그를 읽어서 패턴의 기준 데이터로 저장한다. 저장한 데이터를 이용하여 곡면방정식을 생성하여 위치추적을 수행한다. 리더기에 의해서 인식한 태그의 패턴은 기준데이터가 되는 곡면식에 의해서 리더기의 위치를 파악하게 된다.



<그림 3-9> 위치 추적 시스템의 구축 방법

IV. 결과 및 토의

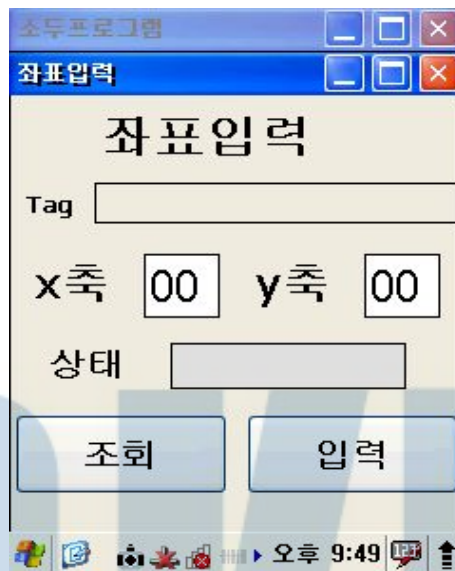
3장에서 제시된 이론의 검증을 위하여 실제 RFID 리더기와 태그들을 가지고 실험을 실시하였다. RFID 리더기는 LS산전의 IU-9060이며, 사양은 900MHz(고주파수 대역) 이동형 리더기이다. 태그는 수동형을 사용하였다. <그림 4-1>은 무선 RFID IU-9060 리더기와 태그의 그림이다.



<그림 4-1> IU-9060 RFID 리더기와 수동형 태그

실험을 위하여 정사각형 그리드 형태의 바닥에 RFID 태그를 300mm 간격으로 가로 9개 세로 9개 씩을 배치하였다. 따라서 생성된 그리드의 노드는 81개이며, 4개의 노드가 이루는 셀의 숫자는 64개이다. 리더기가 태그를 감지하는 면적이 일정하지 않으므로 인식된 태그의 좌표들의 중심을 구하면서 어느 정도 일관성을 유지할 수 있다. 그렇지만 <표 4-1>에서 보는 바와 같이 리더기가 인식하는 위치에 따라 반응하는 태그가 다르게 작동하는 것을 알 수 있다. 이는 환경에 따라서 다르게 나타나기도 하며, 안테나에 잡히는 신호의 세기에 따라 인식하는 태그의 범위가 수시로 변화하기 때문에 나타나는 현상이기도 하다.

<그림 4-2>는 각각의 태그에 좌표를 저장하는 화면이다.



<그림 4-2> 좌표입력 화면

<그림 4-3>는 IU-9060 RFID 리더기로 찍어 <표 4-1>을 측정한 화면이다.



<그림 4-3> 태그 측정 화면

<표 4-1>에서 보는 바와 같이 각 노드에서 감지한 태그의 위치에서 구한 태그의 중심좌표는 실제 좌표와 다른 오차를 포함하고 있다. 다음 <표 4-2>는 실제 위치에 대한 오차를 표시하였다.

<표 4-1> 각 좌표의 중심 값

	1	2	3	4	5	6	7	8	9
1	(2, 1.5)	(2.5, 1.33)	(3.12, 1.38)	(3.78, 1.44)	(4.89, 1.33)	(6, 1.6)	(6.82, 1.64)	(7.83, 1.33)	(7.71, 1.43)
2	(1.82, 2.45)	(2.36, 2.29)	(3.12, 2.12)	(3.29, 2.12)	(5.1, 1.9)	(6.1, 1.8)	(7.14, 1.93)	(7.64, 2.09)	(7.8, 1.9)
3	(1.83, 3.08)	(2.35, 2.88)	(2.85, 2.9)	(4.06, 2.69)	(4.93, 2.8)	(5.94, 3.06)	(6.89, 3.11)	(7.44, 3.06)	(7.92, 3.17)
4	(1.82, 3.91)	(2.33, 3.53)	(2.74, 3.37)	(3.95, 3.81)	(5.1, 4.1)	(5.71, 4.06)	(6.79, 4)	(7.54, 3.62)	(8.08, 3.83)
5	(1.67, 4.67)	(2.38, 4.62)	(3.29, 4.18)	(3.94, 4.39)	(4.69, 4.56)	(5.8, 4.5)	(7.21, 4.79)	(7.6, 4.93)	(8.09, 4.64)
6	(1.88, 5.75)	(2.25, 5.58)	(3.07, 5.73)	(4.42, 5.75)	(5.5, 5.5)	(6.06, 5.69)	(7.21, 5.64)	(7.67, 5.42)	(8, 5.6)
7	(1.71, 6.43)	(2.58, 6)	(3.62, 6)	(3.69, 6.19)	(5.25, 6.58)	(6.57, 6.64)	(7.07, 6.67)	(8.11, 6.33)	(8.29, 7)
8	(1.67, 7.83)	(2.44, 7.67)	(2.8, 7.7)	(4.22, 7.44)	(5.4, 7.4)	(6.57, 7.29)	(7.45, 7.36)	(8, 7.73)	(8.33, 7.83)
9	(1.5, 8.5)	(2, 8.5)	(2.9, 8.3)	(3.91, 8.18)	(5.5, 8.12)	(6, 8.11)	(7.56, 8.33)	(8, 8.5)	(8.5, 8.5)

<표 4-2> 각 좌표 중심 값의 오차

	1	2	3	4	5	6	7	8	9
1	(-1, -0.5)	(-0.5, -0.33)	(-0.12, -0.38)	(0.22, -0.44)	(0.11, -0.33)	(0, -0.6)	(0.18, -0.64)	(0.17, -0.33)	(1.29, -0.43)
2	(-0.82, -0.45)	(-0.36, -0.29)	(-0.12, -0.12)	(0.71, -0.12)	(-0.1, 0.1)	(-0.1, 0.2)	(-0.14, 0.07)	(0.36, -0.09)	(1.2, 0.1)
3	(-0.83, -0.08)	(-0.35, 0.12)	(0.15, 0.1)	(-0.06, 0.31)	(0.07, 0.2)	(0.06, -0.06)	(0.11, -0.11)	(0.56, -0.06)	(1.08, -0.17)
4	(-0.82, 0.09)	(-0.33, 0.47)	(0.26, 0.63)	(0.05, 0.19)	(-0.1, -0.1)	(0.29, -0.06)	(0.21, 0)	(0.46, 0.38)	(0.92, 0.17)
5	(-0.67, 0.33)	(-0.38, 0.38)	(-0.29, 0.82)	(0.06, 0.61)	(0.31, 0.44)	(0.2, 0.5)	(-0.21, 0.21)	(0.4, 0.07)	(0.91, 0.36)
6	(-0.88, 0.25)	(-0.25, 0.42)	(-0.07, 0.27)	(-0.42, 0.25)	(-0.5, 0.5)	(-0.06, 0.31)	(-0.21, 0.36)	(0.33, 0.58)	(1, 0.4)
7	(-0.71, 0.57)	(-0.58, 1)	(-0.62, 1)	(0.31, 0.81)	(-0.25, 0.42)	(-0.57, 0.36)	(-0.07, 0.33)	(-0.11, 0.67)	(0.71, 0)
8	(-0.67, 0.17)	(-0.44, 0.33)	(0.2, 0.3)	(-0.22, 0.56)	(-0.4, 0.6)	(-0.57, 0.71)	(-0.45, 0.64)	(0, 0.27)	(0.67, 0.17)
9	(-0.5, 0.5)	(0, 0.5)	(0.1, 0.7)	(0.09, 0.82)	(-0.5, 0.88)	(0, 0.89)	(-0.56, 0.67)	(0, 0.5)	(0.5, 0.5)

각 위치추적 시스템의 우수성을 평가하기 위하여 임의의 7개 포인트에 대하여 오차를 비교 평가 하였다. <표 4-3>과 같이 7개의 포인트에서 태그를 인식하였으며 리더기에 의해서 얻은 위치 데이터는 <표 4-4>과 같다.

<표 4-3> 실험데이터

(2.5,7.5)									
0	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	5	0	0	0	0	0	0
5	0	0	0	3	0	0	0	0	0
6	5	9	14	0	0	0	0	0	0
7	11	30	29	0	0	0	0	0	0
8	0	29	21	2	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

(2.5,6.5)									
	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	20	0	0	0	0	0	0
5	20	17	17	11	0	0	0	0	0
6	15	23	24	15	6	0	0	0	0
7	13	25	24	4	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

(3.5,5.5)									
	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	7	16	0	0	0	0	0
4	0	0	25	15	11	0	0	0	0
5	0	0	26	24	26	0	0	0	0
6	0	16	26	26	20	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

(4.5,4.5)									
	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	6	0	0
3	0	0	1	15	17	8	0	0	0
4	0	0	13	29	27	15	0	0	0
5	0	0	5	28	26	16	0	0	0
6	0	1	1	1	1	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

(5.5,3.5)									
	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	9	14	19	0	20	0	0
3	0	0	0	20	30	30	30	0	0
4	0	0	0	1	29	30	23	0	0
5	0	0	0	1	1	2	17	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

(6.5,2.5)									
	1	2	3	4	5	6	7	8	9
1	0	0	0	0	3	8	27	0	0
2	0	0	0	0	16	30	29	0	0
3	0	0	0	1	1	29	10	14	0
4	0	0	0	0	12	1	1	8	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

(7.5,2.5)									
	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	3	0	21	10
2	0	0	0	0	1	1	31	25	0
3	0	0	0	0	0	23	31	31	14
4	0	0	0	0	1	0	14	1	15
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

7개의 각 노드에서 인식된 태그에 의해서 중심좌표를 구하였다. <표 4-4>은 각 노드에 대해서 RFID 태그 정보를 획득하여 중심좌표를 구한 표이다.

<표 4-4> 중심 값에 의한 실험 데이터

단위(1/300mm)		
실험 좌표	결과좌표	오차
(2.5,7.5)	(2.55 , 6.55)	0.95
(2.5,6.5)	(2.67 , 6.13)	0.41
(3.5,5.5)	(3.75 , 4.75)	0.79
(4.5,4.5)	(4.41 , 4.35)	0.17
(5.5,3.5)	(5.31 , 3.5)	0.19
(6.5,2.5)	(6.13 , 2.93)	0.57
(7.5,2.5)	(7.2 , 2.6)	0.32

각 노드별 간격을 300mm로 두었으므로 7개의 위치중에서 가장 작은 오차는 좌표(4.5, 4.5)에서 51mm 이며, 가장 큰 오차는 좌표 (2.5, 7.5)에서 285mm 이다. 오차량은 모서리로 갈수록 값이 커지는 것을 알 수 있다. 첫 번째 패턴 매칭에 의한 중심좌표 보다는 좋은 결과를 얻을 수 있었으나, 모서리 부근에서의 오차는 여전히 상대적으로 크게 나타나는 것을 알 수 있다. 평균 오차량은 145.7mm이며, 표준편차는 0.277이다.

각 위치에서의 x축을 중심으로하는 방위각을 계산하였다. 계산 방법은 전 위치와 현재 위치

와의 관계에서 방향벡터를 구하였다. 실제 방향과의 차이는 벡터의 내적에 의하여 구하였다.

<표 4-5> 중심 값에 의한 각도

결과좌표	방위각	오차각
(2.55 , 6.55)	-	-
(2.67 , 6.13)	66.4°	15.9°
(3.75 , 4.75)	51.7°	6.9°
(4.41 , 4.35)	44.6°	13.7°
(5.31 , 3.5)	33.3°	1.6°
(6.13 , 2.93)	25.5°	10.1°
(7.2 , 2.6)	19.8°	17.1°

패턴매칭을 통한 위치추적 방법의 결과이다. <표 4-6>는 각 7개 포인트에서 구한 2진 코드와 각 노드에서 구한 2진 코드를 2진 연산에 의해서 구한 다음 식(6)을 이용하여 매칭비율을 구하였다.

<표 4-6> 2진 연산에 의한 매칭비율

(2,5,7,5)									
	1	2	3	4	5	6	7	8	9
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.143	0.143	0.000	0.000	0.000	0.000	0.000
3	0.143	0.143	0.143	0.000	0.000	0.000	0.000	0.000	0.000
4	0.286	0.143	0.143	0.286	0.143	0.143	0.000	0.000	0.000
5	0.286	0.571	0.429	0.286	0.286	0.000	0.000	0.000	0.000
6	0.857	0.857	1.000	0.286	0.000	0.000	0.000	0.000	0.000
7	0.714	1.000	0.857	0.714	0.143	0.000	0.000	0.000	0.000
8	0.571	0.714	0.714	0.429	0.143	0.000	0.000	0.000	0.000
9	0.143	0.286	0.286	0.286	0.000	0.000	0.000	0.000	0.000

(2,5,6,5)									
	1	2	3	4	5	6	7	8	9
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.077	0.077	0.077	0.077	0.000	0.000	0.000	0.000	0.000
3	0.231	0.231	0.231	0.077	0.000	0.077	0.077	0.000	0.000
4	0.385	0.385	0.385	0.462	0.462	0.231	0.000	0.000	0.000
5	0.308	0.615	0.615	0.538	0.462	0.308	0.077	0.000	0.000
6	0.615	0.846	1.000	0.462	0.231	0.231	0.077	0.000	0.000
7	0.462	0.692	0.692	0.769	0.308	0.154	0.077	0.000	0.000
8	0.154	0.308	0.308	0.154	0.077	0.000	0.000	0.000	0.000
9	0.000	0.000	0.077	0.077	0.000	0.000	0.000	0.000	0.000

(3,5,5,5)									
	1	2	3	4	5	6	7	8	9
1	0.000	0.000	0.000	0.100	0.000	0.000	0.000	0.000	0.000
2	0.100	0.100	0.400	0.400	0.000	0.000	0.000	0.000	0.000
3	0.300	0.400	0.600	0.500	0.400	0.500	0.300	0.100	0.000
4	0.300	0.600	0.800	0.900	0.900	0.700	0.200	0.000	0.000
5	0.100	0.700	1.000	1.000	0.900	0.600	0.200	0.100	0.000
6	0.300	0.500	0.900	0.800	0.400	0.500	0.100	0.000	0.000
7	0.200	0.500	0.700	0.700	0.200	0.100	0.100	0.000	0.000
8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

(4,5,4,5)									
	1	2	3	4	5	6	7	8	9
1	0.000	0.000	0.000	0.091	0.091	0.091	0.091	0.000	0.091
2	0.000	0.000	0.273	0.364	0.273	0.182	0.182	0.182	0.091
3	0.182	0.273	0.545	0.636	0.636	0.818	0.636	0.455	0.182
4	0.182	0.455	0.636	1.000	1.000	1.000	0.545	0.273	0.000
5	0.091	0.364	0.636	0.909	1.000	0.909	0.182	0.182	0.000
6	0.091	0.273	0.455	0.364	0.364	0.455	0.091	0.000	0.000
7	0.000	0.182	0.273	0.273	0.091	0.000	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

(5.5,3.5)									
	1	2	3	4	5	6	7	8	9
1	0.100	0.100	0.200	0.300	0.300	0.300	0.300	0.000	0.100
2	0.100	0.300	0.400	0.500	0.600	0.500	0.400	0.300	0.200
3	0.000	0.300	0.600	0.800	1.000	0.900	0.800	0.700	0.400
4	0.000	0.100	0.200	0.600	0.700	0.900	0.700	0.500	0.300
5	0.000	0.000	0.200	0.400	0.600	0.800	0.300	0.200	0.200
6	0.000	0.000	0.000	0.100	0.200	0.300	0.200	0.200	0.000
7	0.000	0.000	0.200	0.100	0.100	0.100	0.100	0.000	0.000
8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

(6.5,2.5)									
	1	2	3	4	5	6	7	8	9
1	0.000	0.000	0.222	0.333	0.444	0.556	0.556	0.222	0.333
2	0.000	0.111	0.333	0.333	0.667	0.889	1.000	0.556	0.556
3	0.000	0.111	0.222	0.444	0.667	0.889	0.778	0.667	0.333
4	0.000	0.000	0.111	0.333	0.333	0.556	0.556	0.444	0.333
5	0.000	0.000	0.222	0.222	0.222	0.556	0.333	0.222	0.222
6	0.000	0.000	0.000	0.111	0.000	0.111	0.111	0.111	0.111
7	0.000	0.000	0.111	0.000	0.000	0.000	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

(7.5,2.5)									
	1	2	3	4	5	6	7	8	9
1	0.000	0.000	0.000	0.111	0.333	0.556	0.667	0.444	0.556
2	0.000	0.000	0.111	0.111	0.333	0.556	1.000	0.889	0.889
3	0.000	0.000	0.000	0.111	0.444	0.778	1.000	1.000	0.889
4	0.000	0.000	0.000	0.333	0.222	0.444	0.556	0.778	0.667
5	0.000	0.000	0.111	0.111	0.111	0.444	0.444	0.444	0.444
6	0.000	0.000	0.000	0.000	0.000	0.000	0.111	0.222	0.111
7	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
8	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

위 <표 4-6>를 계산식(7)에 의해서 구하면 다음 <표 4-7>와 같다.

<표 4-7> 2진 연산에 의한 실험데이터

단위(1/300mm)		
실험 좌표	결과좌표	오차
(2.5,7.5)	(2.217, 6.783)	0.770
(2.5,6.5)	(2.675, 6.060)	0.473
(3.5,5.5)	(3.814, 4.949)	0.634
(4.5,4.5)	(4.900, 4.017)	0.627
(5.5,3.5)	(5.526, 3.526)	0.037
(6.5,2.5)	(6.600, 2.612)	0.155
(7.5,2.5)	(7.629, 2.546)	0.171

<표 4-7>에서 단위 1은 300mm이므로 최소 오차량은 좌표 (6.5, 2.5)에서 11.1mm이며, 최대 오차량은 좌표 (2.5, 7.5)에서의 231mm 이다. 모서리 근처에서 오차가 크게 나타났으며, 그리드 안쪽에서 좋은 결과를 보임을 알 수 있다. 평균 오차량은 122.87mm이었으며, 표준편차는 0.265로 상대적으로 좋은 결과로 나타났다.

<표 4-8> 2진 연산에 의한 각도

결과좌표	방위각	오차각
(2.217, 6.783)	-	-
(2.675, 6.060)	66.1°	32.3°
(3.814, 4.949)	52.3°	0.7°
(4.900, 4.017)	39.3°	4.3°
(5.526, 3.526)	32.5°	6.8°
(6.600, 2.612)	21.5°	4.6°
(7.629, 2.546)	18.4°	3.6°

태그패턴에 의한 중심좌표 보정 방법을 시험하기 위하여 <표 4-1>을 이용하였다. 비례식을 이용한 위치추적 방법의 결과이다. <표 4-1>을 이용하여 각 7개 포인트의 위치를 찾은 후 비례식을 이용하여 계산하였다. <표 4-9>은 <표 4-1>을 이용하여 찾은 7개의 위치이다.

<표 4-9> <표 4-1>에서 찾은 실험데이터 위치

(2.5,7.5)				
X \ Y	2		3	
	x	y	x	y
7	2.58	6	3.62	6
8	2.44	7.67	2.8	7.7

(2.5,6.5)				
X \ Y	2		3	
	x	y	x	y
7	2.58	6	3.62	6
8	2.44	7.67	2.8	7.7

(3.5,5.5)				
X \ Y	3		4	
	x	y	x	y
5	3.29	4.18	3.94	4.39
6	3.07	5.73	4.42	5.75

(4.5,4.5)				
X \ Y	4		5	
	x	y	x	y
4	3.95	3.81	5.1	4.1
5	3.94	4.39	4.69	4.56

(5.5,3.5)				
X \ Y	4		5	
	x	y	x	y
4	3.95	3.81	5.1	4.1
5	3.94	4.39	4.69	4.56

(6.5,2.5)				
X \ Y	6		7	
	x	y	x	y
2	6.1	1.8	7.14	1.93
3	5.94	3.06	6.89	3.11

(7.5,2.5)				
Y \ X	7		8	
	x	y	x	y
2	7.14	1.93	7.64	2.09
3	6.89	3.11	7.44	3.06

<표 4-9>을 비례식을 이용하여 구한 좌표 및 오차를 구하면 다음 <표 4-10>와 같다.

<표 4-10> 비례식을 이용한 실험데이터

단위(1/300mm)		
실험 좌표	결과좌표	오차량
(2.5,7.5)	(2.057,7.326)	0.476
(2.5,6.5)	(2.229,7.077)	0.638
(3.5,5.5)	(3.570,5.320)	0.194
(4.5,4.5)	(4.489,4.760)	0.260
(5.5,3.5)	(5.364,3.496)	0.136
(6.5,2.5)	(6.111,2.873)	0.539
(7.5,2.5)	(7.352,2.549)	0.155

<표4-10>에서 단위 1은 300mm이므로 최소 오차량은 좌표 (5.5, 3.5)에서 40.76mm이며, 최대 오차량은 좌표 (2.5, 6.5)에서의 191.34mm 이다. 모서리 근처에서 오차가 크게 나타났으며, 그리드 안에서 좋은 결과를 보임을 알 수 있다. 평균 오차량은 102.745mm이었으며, 표준편차는 0.189로 상대적으로 좋은 결과로 나타났다.

<표 4-11> 비례식을 이용한 각도

결과좌표	방위각	오차각
(2.057,7.326)		
(2.229,7.077)	72.5°	34.6°
(3.570,5.320)	56.1°	7.6°
(4.489,4.760)	46.6°	13.6°
(5.364,3.496)	33.0°	10.3°
(6.111,2.873)	25.1°	5.1°
(7.352,2.549)	19.1°	14.6°

B-spline 곡면식을 이용한 위치추적을 실험하기 위하여, 곡면식을 생성하기 위한 데이터 좌표를 <표 4-1>을 이용하여 구하였다. 좌표 데이터는 x,y축에 대하여 9개의 점좌표에 의해서 81개의 좌표 데이터를 이용하여 곡면식을 생성하였다. 곡면식은 81개의 좌표점을 근사곡면으로 하는 5×5 제어점 행렬에 의해서 구하였다.

<표 4-12> 근사곡면으로 하는 5×5 제어점

$\begin{matrix} x \\ y \end{matrix}$	1	2	3	4	5
1	(2.00,1.50)	(2.50,1.30)	(4.57,1.49)	(7.91,1.56)	(7.71,1.43)
2	(1.80,2.69)	(1.56,2.24)	(6.07,2.26)	(7.22,2.49)	(7.92,2.50)
3	(1.79,4.48)	(3.03,4.06)	(3.35,4.32)	(8.17,5.07)	(8.03,4.25)
4	(1.78,7.41)	(2.32,6.44)	(6.02,7.80)	(8.18,6.79)	(8.22,7.76)
5	(1.50,8.50)	(2.00,8.51)	(5.26,7.76)	(8.00,8.49)	(8.50,8.50)

곡면근사에 의해 구한 데이터는 non-uniform한 매개변수식이며, 곡면 전체에 걸쳐서 기울기가 변화하는 곡면식이다. 이 곡면식에 좌표를 대입하여 룩업테이블을 찾아서 비례식을 계산하는 대신에 곡면식에 좌표를 대입하여 좌표를 나타내는 최근접의 매개변수를 구하여 좌표를 구하는 방법으로 위치추적을 수행하면, 다음 <표 4-13>와 같이 결과를 나타냈다.

<표 4-13> B-spline 곡면식을 이용한 실험데이터

단위(1/300mm)		
실험 좌표	결과좌표	오차량
(2.5,7.5)	(2.350,6.845)	0.38
(2.5,6.5)	(2.235,7.253)	0.36
(3.5,5.5)	(3.710,5.334)	0.27
(4.5,4.5)	(4.469,4.797)	0.30
(5.5,3.5)	(5.339,3.707)	0.26
(6.5,2.5)	(6.144,2.930)	0.56
(7.5,2.5)	(7.503,2.440)	0.06

<표 4-13>에서 최대 오차는 좌표 (6.5, 2.5)에서 167.47mm이며, 최소 오차는 좌표 (7.5,2.5)에서의 18.02mm 이다. 대체적으로 비례식에 의해서 구한 좌표와 같이 모서리 근처에서 보다는 그리드 안쪽에서 결과가 더 좋게 나타나는 경향이 있다. 평균 오차는 93.66mm이었으며, 표준편차는 0.139로 나타났다.

<표 4-14> B-spline 곡면식을 이용한 각도

결과좌표	방위각	오차각
(2.350,6.845)	-	-
(2.235,7.253)	72.8°	22.7°
(3.710,5.334)	55.1°	2.8°
(4.469,4.797)	47.0°	13.8°
(5.339,3.707)	34.7°	20.4°
(6.144,2.930)	25.4°	5.1°
(7.503,2.440)	18.0°	6.8°

V. 결 론

실내 위치추적 시스템을 설계하기 위하여 RFID를 이용하여 위치를 추적하는 여러 가지 방법을 시험하여 보았다. 단순히 인식태그의 중심좌표를 구하는 방법은 많은 오차를 포함하고 있으므로 신뢰성을 보장할 수 없다. 태그의 패턴을 인식하여 위치를 추적하는 방법은 잡음을 제거하여 패턴을 매칭하는 방법을 사용하므로 좀 더 발전된 방법으로 정밀도를 높일 수 있는 방법임을 알 수 있다. 그러나 모서리 부분에서는 오차를 완전하게 제거할 수는 없으므로 모든 영역에서 좋은 결과를 얻기는 문제가 발생할 수 있다. 다른 방법으로는 각 노드에서 태그의 인식을 기록한 후, 기록된 데이터에 의해서 오차를 보정하는 방법이 있다. 오차를 보정하기 위해서 비례식을 이용하여 위치를 계산하는 방법과 매개변수 방정식을 이용하여 위치를 계산하는 방법이 있다. 이 두 방법 중에서 매개변수 방정식을 이용하여 위치를 계산하는 방법은 룩업테이블을 생성할 필요가 없으며, 기준 데이터의 생성방법이 간편하여 실제 위치추적 시스템을 구축하는데 적합한 방법이라는 결론을 얻게 되었다.

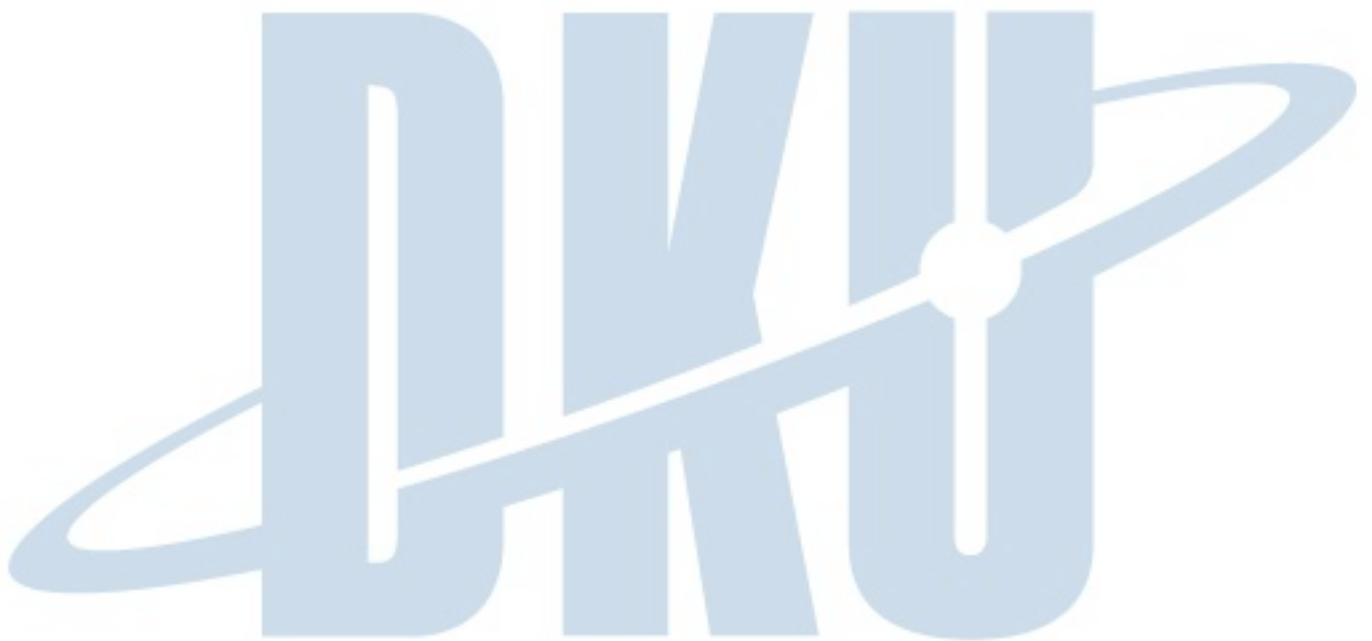
본 연구의 결과로 실내에서 작동하는 RFID 태그를 이용한 저가의 위치 추적 장치를 설계 개발하였다. 개발된 시스템은 반영구적 수동형 RFID 태그를 이용한 시스템이며, Wi-Fi를 이용한 시스템보다 저가이며 상대적으로 높은 정밀도를 보장할 수 있다. 또한, 실내 위치추적 시스템을 실험에 의하여 타당성을 검증하였으며, 시그널의 오작동이나 잡음에 의해 생긴 오차를 보정하는 방법을 제시하였다. 룩업테이블에 의한 보정 대신에 B-spline 곡면식을 보간에 의해 생성한 후 기준 데이터로 사용하였으며, B-spline 곡면 보간에 의해 얻은 결과좌표는 평균 오차가 태그 거리의 28% 이내의 정밀도를 얻을 수 있다. 이 결과는 단순히 좌표의 중심을 구하여 얻는 결과좌표와 비교 하였을 때, 35% 이상의 정밀도 향상을 가져왔다.

향후 연구 과제로는 개발된 실내 위치추적 시스템을 이동로봇, AGV, 스마트폰 위치기반 서비스(LBS) 등에 활용하는 연구가 필요할 것으로 사료된다.

참고문헌

- [1] 김성복, 이상협(2008) 수동 RFID 기반 효율적인 이동로봇 위치 추정. 2008년도 정보 및 제어심포지엄(ICS'08) 논문집, 한국외국어대학교 디지털정보공학부.
- [2] 지용관, 박장현(2009) 수동 RFID Tag 를 기반으로 한 이동 로봇의 경로 계획 알고리즘. 한국정밀공학회지, 한양대학교 자동차공학과.
- [3] 정기호, 장철웅, 강신혁, 이동광, 염문진, 장문석, 공정식, 권오상, 이웅혁(2007) 실내 환경에서 RFID와 초음파를 이용한 이동로봇의 위치 추정에 관한 연구.
- [4] 정기호, 장철웅, 심현민, 장승관, 이웅혁(2006) RFID 파워 컨트롤을 이용한 이동로봇의 위치 추정에 관한 연구. 2006년도 정보 및 제어 학술대회.
- [5] 김원(2007) RFID 환경 기반 이동 로봇 시스템. 한국정보기술학회논문지, 우송대학교 컴퓨터정보학과.
- [6] 남상엽(2009), 컨버전스 RFID/USN 임베디드 시스템. jinhan M&B.
- [7] 윤성길(2005), 알기쉬운 RFID. 도서출판 미래컴. PP 29~35
- [8] 김갑용(2005), 유비쿼터스 센서 네트워크 기술. jinhan M&B, PP 246~249
- [9] 조형민, 이정우(2010) 외부 GPS 모바일 단말기를 이용한 실내 위치 추적 기법. 한국방송공학회, 서울대학교 전기공학부.
- [10] F A. and Retscher, G., "Active RFID Trilateration and Location Fingerprinting Based on RSSI for Pedestrian Navigation", Journal of Navigation, 62(2), pp. 323-340, 2009.
- [11] Mori, T., Siridanupath, C., Noguchi, H., and Sato, T., "Active RFID-based indoor object management system in sensor-embedded environment", Proceedings of the 5th International Conference on Networked Sensing Systems 2008, Kanazawa, Japan, 17-19, pp. 224, 2008.
- [12] Zhou, Y., Liu, W., and Huang, P., "Laser-activated RFID-based Indoor Localization System for Mobile Robots, Proceedings of the 2007 IEEE International Conference on Robotics and Automation, pp. 4600-4605, 2007.

- [13] Wang, H.P., Hewgill, D.E., and Vickers, G.W., "An Efficient Algorithm for Generating B-Spline Interpolation Curves and Surfaces from B-Spline Approximations", Communications in Applied Numerical Methods, Vol. 6, pp395-400, 1990.

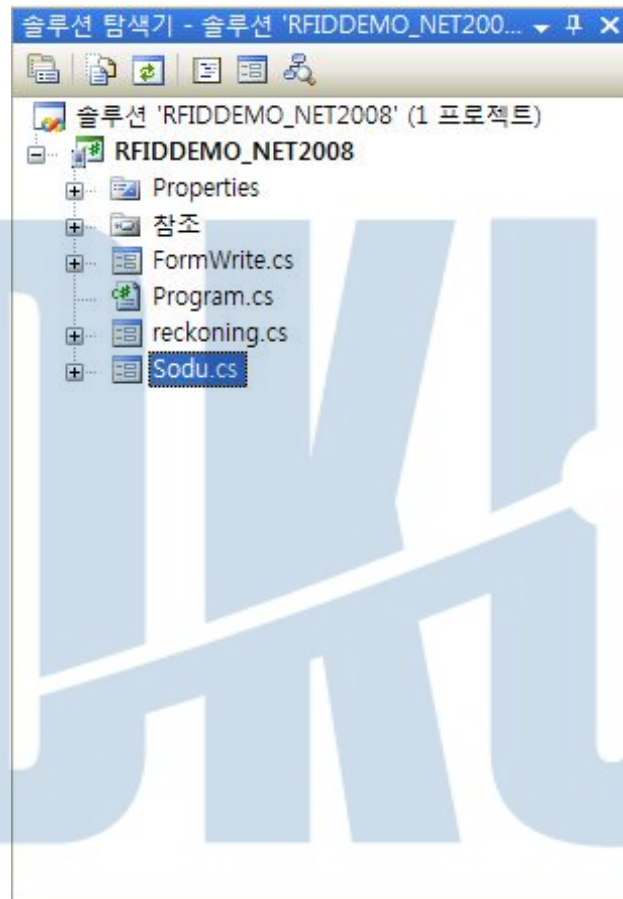


<부록 1> 81개 데이터 2진 노드

(2,7)	11111000011100000011100000011000000010000000000000000000000000000000000000
(2,2)	11111000011111000011110000001100000010000000000000000000000000000000000000
(2,3)	11111000011111000011110000001100000010000000000000000000000000000000000000
(2,4)	0111100001111100000011100000011100
(2,5)	0001110000011110000011110000001100
(2,6)	000111100000011100000111000000010100
(2,7)	00000110000001111000101110000000010000001000000000000000000000000000000000
(2,8)	00000011000000011100000011000000011000000000000000000000000000000000000000
(2,9)	000000011000000011000000011000
(3,1)	10000000110000001100000001000000011000000000000000000000000000000000000000
(3,2)	11100000011100000011110000011110000011000000010000000000000000000000000000
(3,3)	11111000001110000011111000011110000001110000000000000000000000000000000000
(3,4)	01111000011111000010111000010111000000110000000000000000000000000000000000
(3,5)	00011100000011000000111100010111000000111000100000000000000000000000000000
(3,6)	00000110000001110000011110000011110000011000000000000000000000000000000000
(3,7)	0000010000000011000000011100010111000001010000000000000001000000000000000000
(2,8)	00000011000000011000000011100000011000000010000000000000000000000000000000

<부록 2 > PDA 위치 추적 C# 프로그램

PDA위치 추적 시스템 프로그램의 언어로는 Microsoft의 Microsoft Visual Studio 2008을 사용하였으며 그림은 Workspace를 나타내었고 프로그램의 Code는 전부 표시할 수 없어 메인이라 할 수 있는 Sodu.cs만 수록하였다.



```

using System;
using System.Runtime.InteropServices;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Media;
using System.IO;

namespace RFIDDEMO_NET
{
    public partial class RFIDDEMO : Form
    {
        public class MKLibU
        {
            [DllImport("IU906xRFID.dll")]
            public static extern int IUInstanceOpen();
            [DllImport("IU906xRFID.dll")]
            public static extern void IUInstanceClose();
            [DllImport("IU906xRFID.dll")]
            public static extern int IUConnect(string strPort);
            [DllImport("IU906xRFID.dll")]
            public static extern void IUDisconnect();
            [DllImport("IU906xRFID.dll")]
            public static extern int IUSetFrequency(double dFreqStart, double dFreqEnd, double dFreqStep, int modeFreq);
            [DllImport("IU906xRFID.dll")]
            public static extern int IUBeginReadTag();
            [DllImport("IU906xRFID.dll")]
            public static extern int IUFinishReadTag();
            [DllImport("IU906xRFID.dll")]
            public static extern int IUGetTagUIDCount();
            [DllImport("IU906xRFID.dll")]
            public static extern int IUGetTagReadCount(int nIndex);
            [DllImport("IU906xRFID.dll")]
            public static extern int IUGetTagReadTime(int nIndex);
            [DllImport("IU906xRFID.dll")]
            public static extern int IUGetTagUID(int nIndex, byte[] lpTagUID);
            [DllImport("IU906xRFID.dll")]
            public static extern int IUGetTagRSSI(int nIndex, int[] lpTagRSSI);
            [DllImport("IU906xRFID.dll")]
            public static extern void IUDeleteAllDB();
            [DllImport("IU906xRFID.dll")]
            public static extern IntPtr IUGetReaderVersion();
        }
    }
}

```

```

public static extern IntPtr IULibVersion();
[DllImport("IU906xRFID.dll")]
public static extern void IUReaderPowerOn(int powerMode);
[DllImport("IU906xRFID.dll")]
public static extern void IUReaderPowerOff();
[DllImport("IU906xRFID.dll")]
public static extern int IUGetReaderType();
}

public RFIDDEMO()
{
    InitializeComponent();

    MKLibU.IUInstanceOpen();

    InitializeProgram();
}

private void InitializeProgram()
{
    MKLibU.IUReaderPowerOn(0);

    if (MKLibU.IUConnect("COM4:").Equals(0)) MessageBox.Show("Error to Connect Reader");

    int nReaderType = MKLibU.IUGetReaderType();

    if(nReaderType == 0 || nReaderType == 1) MKLibU.IUSetFrequency(917.3, 920.3, 0.6, 1);
    else if (nReaderType == 2) MKLibU.IUSetFrequency(865.70, 867.50, 0.6, 2);
    else if (nReaderType == 3) MKLibU.IUSetFrequency(952.4, 953.6, 0.2, 2);
    else if (nReaderType == 4) MKLibU.IUSetFrequency(903.20, 927.70, 0.5, 1);
    else if (nReaderType == 5) MKLibU.IUSetFrequency(920.625, 924.375, 0.25, 1);
    else if (nReaderType == 6) MKLibU.IUSetFrequency(865.2, 867.8, 0.2, 1);
    else if (nReaderType == 7) MKLibU.IUSetFrequency(918.2, 919.8, 0.2, 1);
    else if (nReaderType == 8) MKLibU.IUSetFrequency(919.2, 922.8, 0.2, 1);
    else if (nReaderType == 9) MKLibU.IUSetFrequency(920.2, 924.8, 0.2, 1);
    else MessageBox.Show("Error Reader Type");

    buttonStop.Enabled = false
}

private void OnDestroy(object sender, EventArgs e)
{
    MKLibU.IUFinishReadTag();
    MKLibU.IUDisconnect();

    MKLibU.IUReaderPowerOff();

    MKLibU.IUInstanceClose();
}

```

```

private void buttonVersion_Click(object sender, EventArgs e)
{
    string strVersion = Marshal.PtrToStringUni(MKLibU.IUGetReaderVersion());

    MessageBox.Show(strVersion);
}

private void buttonRead_Click(object sender, EventArgs e)
{
    buttonStop.Enabled = true
    buttonRead.Enabled = false
    buttonRW.Enabled = false

    MKLibU.IUBeginReadTag();
    timerTagPrint.Enabled = true

    timer1.Enabled = true
}
private void timerTagPrint_Tick(object sender, EventArgs e)
{
    OnTagUidOutput();
}

private void OnTagUidOutput()
{
    string strTagUID;
    int nTagReadCount;

    int nTagCount = MKLibU.IUGetTagUIDCount();
    if (nTagCount.Equals(0)) return

    byte[] m_lpUIDData = new byte[128];
    int[] lpUIDRSSI = new int[10];

    for (int nTag = 0; nTag < nTagCount; nTag++)
    {
        int nTagUIDSize = MKLibU.IUGetTagUID(nTag, m_lpUIDData);

        string strTagUIDHex = null
        int nHexValue;

        for (int nI = 0; nI < nTagUIDSize; nI++)
        {
            nHexValue = Convert.ToInt32(m_lpUIDData[nI]);

```

```

strTagUIDHex += string.Format("{0:X2}", nHexValue);
}

strTagUID = strTagUIDHex;

nTagReadCount = MKLibU.IUGetTagReadCount(nTag);

MKLibU.IUGetTagRSSI(nTag, lpUIDRSSI);

InsertTagData(strTagUID, nTagReadCount, lpUIDRSSI[0]);
}

if (listViewTagUID.Items.Count.Equals(0)) return
else labelTagCount.Text = this.listViewTagUID.Items.Count.ToString();

PlayReadingSount();

MKLibU.IUDeleteAllIDB();
}

private void InsertTagData(string strTagUID, int nTagReadCount, int nRSSI)
{
    string strTagReadCount;
    string strListTagUID;
    int nListSize = listViewTagUID.Items.Count;

    int nMidKey = 0;
    int nLeftKey = 0;
    int nRightKey = nListSize - 1;

    int nResult = 1;

    while (nRightKey >= nLeftKey)
    {
        nMidKey = (nLeftKey + nRightKey) / 2;

        strListTagUID = listViewTagUID.Items[nMidKey].SubItems[0].Text;

        nResult = String.Compare(strTagUID, strListTagUID);

        if (nResult < 0) nRightKey = nMidKey - 1;
        else nLeftKey = nMidKey + 1;

        if (nResult == 0) break
    }

    if (nResult != 0)
    {
        int nInsertPoint = nMidKey + ((nResult > 0) ? 1 : 0);
    }
}

```

```

if (nInsertPoint > nListSize) nInsertPoint = nListSize;

strTagReadCount = nTagReadCount.ToString();

InsertTagList(nInsertPoint, strTagUID, strTagReadCount, nRSSI.ToString());
}
else
{
int nCurTagCount = int.Parse(this.listViewTagUID.Items[nMidKey].SubItems[1].Text) + 1;

strTagReadCount = nCurTagCount.ToString();

SetTagCount(nMidKey, strTagReadCount, nRSSI.ToString());
}
}

private void InsertTagList(int nInsertPoint, string strTagUID, string strTagReadCount, string
strRSSI)
{

ListViewItem lvi;
ListViewItem.ListViewSubItem lvsi;

lvi = new ListViewItem();
lvi.Text = strTagUID;

lvsi = new ListViewItem.ListViewSubItem();
lvsi.Text = "1"
lvi.SubItems.Add(lvsi);

lvsi = new ListViewItem.ListViewSubItem();
lvsi.Text = strRSSI;
lvi.SubItems.Add(lvsi);

this.listViewTagUID.Items.Insert(nInsertPoint, lvi);
this.listViewTagUID.EnsureVisible(this.listViewTagUID.Items.Count - 1);
}

private void SetTagCount(int nItem, string strTagReadCount, string strRSSI)
{
this.listViewTagUID.Items[nItem].SubItems[1].Text = strTagReadCount;
this.listViewTagUID.Items[nItem].SubItems[2].Text = strRSSI;

this.listViewTagUID.EnsureVisible(this.listViewTagUID.Items.Count - 1);
}

private void buttonStop_Click(object sender, EventArgs e)
{

```

```

timerTagPrint.Enabled = false
MKLibU.IUFinishReadTag();

buttonStop.Enabled = false
buttonRead.Enabled = true
buttonRW.Enabled = true
}

private void buttonClear_Click(object sender, EventArgs e)
{
    this.listViewTagUID.Items.Clear();
    this.labelTagCount.Text = "0"
}

private void PlayReadingSount()
{
    SoundPlayer simpleSound = new SoundPlayer(@"\Program Files\RFIDDEMO_NET\reading.wav");
    simpleSound.Play();
}

private void buttonRW_Click(object sender, EventArgs e)
{
    FormWrite frmReadWrite = new FormWrite();
    frmReadWrite.ShowDialog();
}

private void button1_Click(object sender, EventArgs e)
{
    File.Delete("좌표");
    int i;

    for (i = 0; i < this.listViewTagUID.Items.Count; i++)
    {
        String strTemp = listViewTagUID.Items[i].SubItems[0].Text.Substring(4, 2);
        String strTemp1 = listViewTagUID.Items[i].SubItems[0].Text.Substring(6, 2);
        String sBuffer = "x : " + strTemp + "    " + "y : " + strTemp1 + "\r\n";
        FileInfo fileInfo = new FileInfo("좌표");
        FileStream fileStream = fileInfo.Open(FileMode.OpenOrCreate, FileAccess.ReadWrite, FileShare.Read);
        StreamWriter streamWriter = new StreamWriter(fileStream); //,
        streamWriter.BaseStream.Seek(0, SeekOrigin.End);
        streamWriter.Write(sBuffer);
        streamWriter.Flush();
        streamWriter.Close();
        fileStream.Close();
    }
}

```



```

StreamReader sr = new StreamReader("좌표", Encoding.Default);
textBox1.Text = sr.ReadToEnd();
sr.Close();

}

private void timer1_Tick(object sender, EventArgs e)
{
    timerTagPrint.Enabled = false
    MKLibU.IUFinishReadTag();
    buttonStop.Enabled = false
    buttonRead.Enabled = true
    buttonRW.Enabled = true
    timer1.Enabled = false
}

private void button2_Click(object sender, EventArgs e)
{
    int i;
    String a = textBox2.Text;
    String b = textBox3.Text;
    String c = x.Text;
    String d = y.Text;

    for (i = 0; i < this.listViewTagUID.Items.Count; i++)
    {
        String strTemp = listViewTagUID.Items[i].SubItems[0].Text.Substring(4, 2);
        String strTemp1 = listViewTagUID.Items[i].SubItems[0].Text.Substring(6, 2);
        String strTemp2 = listViewTagUID.Items[i].SubItems[1].Text;
        String sBuffer = "x : " + strTemp + " " + "y : " + strTemp1 + " " + "n : " + strTemp2
        + "\r\n"
        FileInfo fileInfo = new FileInfo( a+ c + b + d + ".txt");
        FileStream fileStream = fileInfo.Open(FileMode.OpenOrCreate, FileAccess.ReadWrite,
        FileShare.Read);
        StreamWriter streamWriter = new StreamWriter(fileStream);
        streamWriter.BaseStream.Seek(0, SeekOrigin.End);
        streamWriter.Write(sBuffer);
        streamWriter.Flush();
        streamWriter.Close();
        fileStream.Close();

    }

    String sBuffer1 = "\r\n"

    FileInfo fileInfo1 = new FileInfo(a + c + b + d + ".txt");
    FileStream fileStream1 = fileInfo1.Open(FileMode.OpenOrCreate, FileAccess.ReadWrite, FileShare.Read);
    StreamWriter streamWriter1 = new StreamWriter(fileStream1);
        streamWriter1.BaseStream.Seek(0, SeekOrigin.End);

```

```

streamWriter1.Write(sBuffer1);
streamWriter1.Flush();
streamWriter1.Close();
fileStream1.Close();

}

private void button3_Click(object sender, EventArgs e)
{
    int i;
    int ax = 0;
    int ay = 0;

    for (i = 0; i < this.listViewTagUID.Items.Count; i++)
    {

        int strTemp3 = Convert.ToInt32(listViewTagUID.Items[i].SubItems[0].Text.Substring(4, 2));
        int strTemp4 = Convert.ToInt32(listViewTagUID.Items[i].SubItems[0].Text.Substring(6, 2));
        int strTemp5 = Convert.ToInt32(listViewTagUID.Items[i].SubItems[1].Text);

        ax += strTemp3;
        ay += strTemp4;

    }

    Double avx = ax / this.listViewTagUID.Items.Count;
    Double avy = ay / this.listViewTagUID.Items.Count;
    Double avx1 = Math.Round(avx, 2);
    Double avy1 = Math.Round(avy, 2);
    string avx2 = Convert.ToString(avx1);
    string avy2 = Convert.ToString(avy1);
    textBox5.Text = "(" + avx2 + " , " + avy2 + ")"

}

private void button4_Click(object sender, EventArgs e)
{
    reckoning math = new reckoning();
    math.ShowDialog();
}
}
}

```

(Abstract)

Research on Position Tracking Using Passive RFID Tags

Lee, Chang Hwan

Department of Industrial Engineering

Graduate School

Dankook University

Advisor : Professor Cho, Jae Hyung

In modern technology, wireless positioning system can be used in mobile robot tracking systems, the Internet's location-based services (LBS) applications and so on. The position of the tracking system, GPS system, the most known systems, or to be used in outdoor locations through satellite tracking system. However, the GPS receiver indoors was not working properly.

Include an indoor location tracking system, the Bluetooth and Wi-Fi Positioning System relies on the use of environmental technologies, and these devices show a precision of about 1 ~ 10m. With this system, to track the location and time investment, not cost-effective conditions are not varying degrees of success. RFID systems for indoor LBS system has required the introduction of cost-effectiveness and to ensure the accuracy of the device in indoor localization is the development time required.

However, most of the active recognition system using active RFID tags, active

tags with batteries using the method should be used. Using passive RFID tags to track the location of the interior systems are facing problems in precision and accuracy.

In this paper, we estimate a low-cost indoor positioning system using the RFID system design was developed. The indoor location tracking to allow more precise tracking algorithm is applied to various locations, the results were compared.

As a result of this study using RFID tags that operate in an indoor location tracking devices, low-cost design has been developed. The developed system is a system using a semi-passive RFID tags. The system is relatively more low-cost and high precision can be guaranteed. In addition, the indoor location tracking system was tested by validating the signal to noise caused by the malfunction or error how to calibrate presented. Correction due to the lookup table, instead of B-spline surfaces generated by the interpolation formula was used as baseline data. This system can be further utilized for a mobile robot system, AGV, phone location-based services (LBS), etc.