Journal of The Korea Society of Computer and Information
Vol. 19, No. 11, November 2014
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2014.19.11.025

스마트폰을 활용한 지능형 로봇 SLAM 기법의 오버헤드 개선방안에 관한 연구

이 철 원*, 전 흥 석*

A Study on Improving the Computational Complexity of SLAM for Intelligent Robot Utilizing Smart Phone

Cheol-Won Lee*, Heung-Seok Jeon*

요 약

본 논문에서, 우리는 지능형 로봇의 계산적 슬램 오버헤드를 개선하는 새로운 계획을 제안한다. 아이디어는 스마트폰의 유휴자원을 하나의 슬램 프로세서처럼 사용하는 것이다. 우리는 로봇과 스마트폰이 협력하는 하나의 새로운 모델을 디자인하였다. 실험결과로, 스마트폰이 슬램의 성능을 향상시키는 것에 매우 도움이 되고, 로봇이 더 빠르게 움직일 때, 스마트폰으로 인한 이득이 증가하는 것을 발견했다.

▸ Keywords : 로봇, 슬램, 스마트폰, 계산적 오버헤드

Abstract

In this paper, we propose a new scheme for enhancing the computational SLAM overhead of intelligent robots. The idea is to use the idle resource of Smart Phone as a SLAM processor. We designed a new model for incorporating the smart phone with robot. According to the experimental results, we found that the smart phone was very helpful for improving the SLAM performance and the gain from the smart phone was increased as robot moves faster.

▸ Keywords : Robot, SLAM, Smart Phone, Computational Overhead

•제1저자 : 이철원 •교신저자 : 이철원
•투고일 : 2014. 3. 27, 심사일 : 2014. 8. 19, 게재확정일 : 2014. 10. 22.
* 건국대학교 컴퓨터공학과(Dept. of Computer Engineering, Kunkuk University)

I. 서 론

유SLAM(Simultaneous Localization And Mapping)이란 로봇이 미지의 환경을 돌아다니면서 로봇에 부착되어 있는 센서만으로 외부의 도움 없이 환경에 대한 정확한 지도를 작성함과 동시에 자신의 위치를 알아내는 기법으로 로봇의 자율주행을 위한 핵심기술이다. SLAM은 Mobile Robotics Community의 상징적인 연구 주제가 되어왔고, 로봇을 정말로 자율적으로 만드는 것의 수단으로 제공되어왔다[1],[10].

더 정확한 지도와 로봇의 위치를 알아내기 위해서 다양한 SLAM 기법들이 개발되어지고 있다. Extended Kalman Filter(EKF)를 사용하여 데이터 연결의 최대 우도 알고리즘을 적용한 EKF-SLAM[2], 모든 랜드 마크들을 유지하는데 발생하는 오버헤드를 줄이기 위해서 필요한 랜드 마크만 유지하는 기법의 FastSLAM[3], 랜드 마크 없이 로봇의 위치와 맵을 유지하기 위해서 파티클 필터를 사용하는 DP-SLAM[4],[9] 외에도 다양한 SLAM 알고리즘들이 연구되고 있다.

그럼에도 불구하고 SLAM은 아직 현대의 지능형 로봇에는 많이 사용되고 있지 않다. 그 이유 중 한 가지는 로봇에서 발생하는 Computational 오버헤드가 매우 크다는 것이다[5],[8].

DP-SLAM을 기준으로 설명하자면, 로봇이 주행하면서 센서로부터 입력받는 환경 데이터를 기반으로 맵과 로봇의 정보 트리 구조를 유지 및 업데이트 한다. 로봇의 이동속도가 증가할수록 동일한 맵을 만들어 내기위해서 동일한 시간동안에 더 빠른 파티클의 비교연산이 필요하고 이것은 SLAM의 Computation-al 오버헤드 증가로 나타난다. 로봇의 성능은 제한되어 있기 때문에 로봇이 처리할 수 있는 속도이상으로 빨라지는 경우 발생하는 Computational 오버헤드를 처리하지 못하게 되고, 맵 품질 저하가 결과로 나타난다.

그림 1에서는 로봇의 이동속도 증가에 따른 맵의 완성도의 변화를 보여준다. 본 실험에서 사용한 로봇의 특정 위치로 이동명령을 수행하여 유효한 맵을 만들어 낼 수 있는 임계속도는 0.036m/sec이다. 그 이상의 이동속도에서는 Computational 오버헤드를 처리하지 못하여 맵의 완성도가 떨어지는 것을 그림1 (c), (d)에서 확인할 수 있다.

이러한 문제를 해결하기 위해서 본 논문에서는 스마트폰을 활용하고자 한다. 최근 스마트폰 성능의 비약적인 향상으로 스마트폰에서 다양한 컨텐츠를 즐길 수 있게 되었다. 2013년 하루 평균 스마트폰 이용시간은 66분으로 조사되었다. 하루 24시간 중 평균적으로 66분을 제외한 시간은 스마트폰의 대기시간 즉 유휴시간이라는 것이다. 이러한 유휴시간에 스마트폰의 자원을 활용한다면 로봇의 하드웨어 업그레이드 없이 이동속도 증가에 따른 오버헤드를 개선시킬 수 있다고 생각한다.

따라서 본 논문에서는 로봇이 유휴시간중인 스마트폰의 자원을 활용하여 로봇의 이동속도에 비례하여 증가하는 SLAM 알고리즘의 오버헤드를 개선하고, SLAM을 사용하는 로봇의 이동속도를 증가시킬 수 있는 방안을 제안하였다.

본 논문은 다음과 같이 구성된다. 2절에서는 스마트폰의 자원공유를 이용하여 로봇의 오버헤드를 개선하기위한 아이디어 모델을 제시한다. 3절에서는 2절의 아이디어를 평가하기 위한 맵 평가기법인 PMC 지수를 제안한다. 4절은 2절에서 제시한 아이디어 모델을 구현하기 위한 사전 실험들과 본 모델의 성능실험 그리고 평가 결과를 보여준다. 마지막으로 5절은 본 논문의 결론을 맺는다.

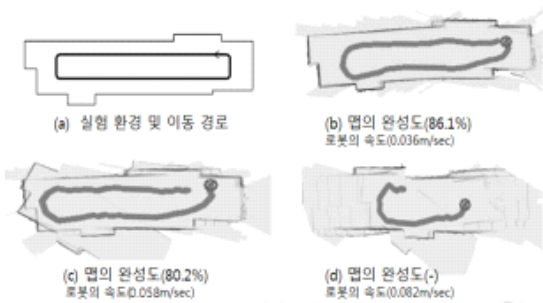


그림 1. 로봇 이동속도에 따른 맵의 완성도
Fig. 1. Map completeness in each robot speed

II. 스마트폰의 자원공유를 통한 로봇의 오버헤드 개선 모델

본 논문에서는 스마트폰이 자동 잠금 상태가 되어 대기 중인 시간을 유휴시간이라고 정의한다. 이러한 유휴시간중인 스마트폰을 활용하여 프로세싱 자원을 통신으로 공유할 수 있다. 그러므로 공유된 자원에서 SLAM의 연산을 할 수 있다면 SLAM 알고리즘을 사용하는 로봇들의 성능적 한계치를 추가적인 하드웨어 업그레이드 없이 오버헤드를 개선 할 수 있다.

로봇에서 SLAM 알고리즘이 실행되는 순서를 간략히 설명하자면, 먼저 로봇이 센서를 통해 환경에 대한 정보를 입력받는다. 다음으로 SLAM Task를 통해 맵과 로봇의 위치 그리고 방향 정보를 유지한다[1]. 구축된 맵과 로봇의 정보를

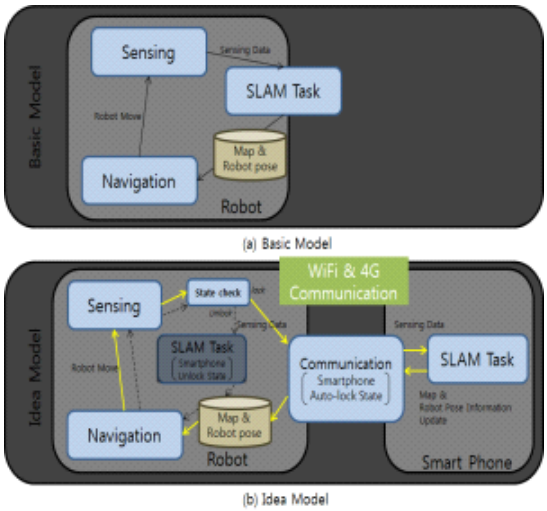


그림 2. Basic Model과 Idea Model의 흐름도
Fig. 2. Basic model and Idea model flowchart

통해서 로봇이 움직여야하는 경로를 만들고 움직인다. 이후 다시 환경정보를 센서로 입력받는 순환을 한다. 이러한 순환을 그림 2 (a)의 Basic Model에서 볼 수 있다.

SLAM Task는 이동속도에 증가했을 때, 완성도를 가진 맵을 만들어내기 위해서 연산량이 가파르게 상승하게 되고, Com-putational 오버헤드 문제가 발생하게 된다. 이유로는, SLAM Task를 통해 구축되어 있는 맵과 센서로부터 들어오는 환경의 정보를 비교하는 작업을 반복하게 되는데 로봇의 이동속도가 증가하게 되면 이 반복적인 작업을 더 빠르게 수행해야만 맵과 로봇의 위치 및 방향정보를 정확하게 갱신하게 된다. 만약 로봇의 이동속도를 SLAM Task의 처리속도가 따라가지 못한다면, 비교하는 작업의 오버헤드를 처리하는 동안, SLAM Task에서 비교할 수 있는 환경의 범위 이상의 거리를 이동하기 때문에 SLAM Task가 진행될수록 구축된 맵과 실제 환경은 매우 큰 차이를 발생시킨다. 결과로, 연산되지 못한 Task들로 인한 오류가 많은 상태에서 만들어진 SLAM 결과맵의 신뢰도 또한 보장할 수 없다.

본 논문에서는 제한된 성능의 로봇에서 SLAM 알고리즘을 동작하는 속도를 증가시키기 위해, 유휴시간중인 스마트폰 자원을 이용하여 문제를 해결할 수 있는 Idea Model을 제시하고 전체적인 흐름은 그림 2. (b)Idea Model과 표 1. 스마트폰을 활용한 슬램 알고리즘에서 확인할 수 있다.

제시된 Idea Model을 설명하자면, 로봇은 그림 2 (a)의 Basic Model의 순환 도중, State check 모듈을 통해 스마트폰의 상태를 확인한다. 스마트폰이 사용 중이라면 기존의

표 1. 스마트폰을 활용한 슬램 알고리즘
Table 1. Algorithm SLAM by Utilizing Smart Phone

Algorithm : SLAM by Utilizing Smart Phone
Input : Sensing data
Output : Map, Robot pose information
BEGIN
1 . initialize variables in SLAM
2 . LOOP
3 . Sensing in environment (Laser 181, Odometer R/L)
4 . CALL Statecheck(output is Lock(1) or UnLock(0))
5 . IF Statecheck output is 1
6 . CALL Communication(Send Sensing data)
7 . Smart Phone SLAM Task On/ Robot SLAM Task Off
8 . CALL Communication(Receive Map & Robot pose)
7 . ENDIF
9 . IF Statecheck output is 0
10. Robot SLAM Task run
11. ENDIF
12. UPDATE Map & Robot pose DB
13. CALL Navigation(Navigation and Robot move refer Map & Robot pose DB)
14. ENDLOOP
END

Basic Model의 순환을 수행하고, 유휴시간중이라면 스마트폰으로 수집한 데이터를 전송한다. 이 때, 로봇의 SLAM Task는 비활성화 되며, 스마트폰에서는 전송받은 데이터로 SLAM Task를 연산하여 맵과 로봇의 위치 그리고 방향정보를 결과물로 얻는다. 스마트폰은 로봇으로 스마트폰 SLAM Task의 결과물을 전송하고 로봇에서는 저장되어있는 정보를 업데이트 한다. 로봇은 업데이트된 맵과 로봇의 위치 정보를 토대로 만들어진 이동경로에 따라 로봇의 이동 명령, 즉 바퀴의 Odometer value를 전송한다. 그리고 다시 센서를 통해 환경의 정보를 수집하는 단계로 순환을 한다.

위에서 소개한 Idea Model은 본 논문에서는 DP-SLAM을 기반으로 설명하지만, 프로세스의 모듈단위 개선이기 때문에 다른 SLAM 알고리즘을 사용하는 로봇에 적용할 수 있다. 오버헤드가 과도하게 발생하는 SLAM Task를 스마트폰에서 처리하게 된다면, 오버헤드로 인해 발생한 오류들이 줄어들 것이며, SLAM의 결과맵의 정확도와 로봇의 위치 그리고 방향 또한 더 정확하게 보정될 것을 기대할 수 있다. 또한 연산능력 부족으로 인해 할 수 없었던 SLAM 외에 다른 응용작업도 추가로 할 수 있을 것이다.

III. Pixel Map Completeness 지수

본 논문에서는 로봇 이동속도에 따른 SLAM의 결과맵의

완성도를 통해서 오버헤드 개선에 대한 평가를 진행할 것이다. 이에 적합한 SLAM의 결과맵을 평가할 수 있는 방법으로 맵에 대한 완성도를 픽셀 단위로 계산하여 퍼센트로 표기하는 Pixel Map Completeness(PMC)지수를 제안한다.

PMC지수를 계산하는 방법은 다음과 같다. 그림 3 (a)와 같이 실제 실험 환경을 측정하여 측정된 실제 환경을 비율로 나누어 맵을 만든다(이하 비율도라고 칭한다). 그림 3 (b)의 비율도 내부는 픽셀로 나누어지며 2진수로 배열에 저장된다. 그림3의 (b)와 (c)의 픽셀은 가상의 픽셀이며, 실제로는 $1m^2$ 당 100개의 픽셀로 나누어진다. 이후 비율도와 DP-SLAM의 결과맵에서 가장 왼쪽 상단에 있는 외곽선을 시작점으로 추출한다. 추출된 외곽선 시작점을 저장되어있는 배열별로 시작점을 겹쳐서 비교한다. 그림 3 (c)는 실험 환경의 실측된 환경의 비율도와 실험환경의 DP-SLAM 결과 맵의 시작점을 겹치고, 내부를 픽셀 단위로 비교 하는 것을 보여준다. 비율도와 비교된 DP-SLAM의 결과 맵은 PMC 98.3%의 결과 값을 가지며 1.7%의 손실율을 보여주고 있다.

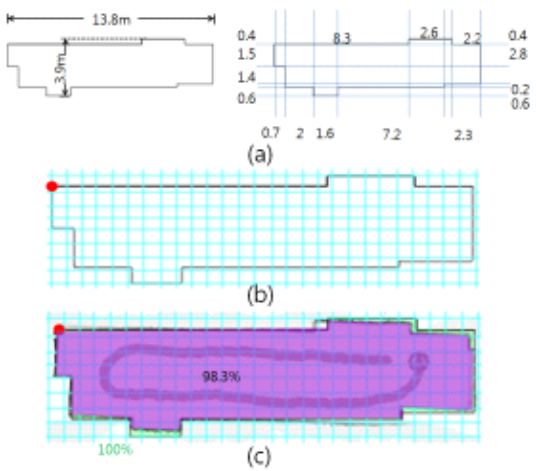


그림 3. PMC기법의 예시 (a) 실제 환경의 비율도 (b) 비율도의 픽셀화 예시 180픽셀 PMC 100% (c) 비율도와 DP-SLAM의 결과맵 비교 177픽셀 PMC 98.3%
Fig. 3. PMC example (a) real environment ratio map (b) PMC 100% map (180 pixel) (c) real map and DP-SLAM result map comparison (PMC 98.3%)

표 2. PMC지수 등급 구분표
Table 2. PMC value rating

등급	구간	의미
PMC-AM	100~90%	평가된 맵에서 명령을 정확하게 수행할 수 있다.
PMC-BM	89~85%	평가된 맵에서 명령을 수행할 수는 있으나 시간이 흐름에 따라 오류가 발생할 수 있다.
PMC-CM	84~ 0%	평가된 맵에서 명령을 수행할 수 없다.

측정된 PMC지수는 수치별로 표 2와 같은 등급으로 구분한다. 본 논문에서 실험한 실험환경에서 실험된 PMC 지수를 통해서 등급을 구분하였으며, 실험환경에 따라서 오차가 발생할 수 있다.

IV. Experiment

1. 실험에 사용된 로봇의 구현

실험에 사용된 로봇 bot의 제원은 다음과 같다. 프레임의 원판의 지름이 36cm이고, 바퀴는 원판의 중앙에서 각각 15cm 떨어진 부분에 위치한 지점에 배치하였다. 바퀴의 두께는 2.5cm이고, 홈은 직사각형 모양으로 14cm×3.5cm이다. 로봇의 프레임은 나무로 제작되었으며, 그림 4와 같이 2단부로 구성되어 있다. 1단은 바퀴와 엔코터 부착형 모터, 제어보드 그리고 레이저 센서가 위치한다. 2단은 노트북 컴퓨터와 14.8V 배터리가 위치한다. 로봇의 전원은 배터리에 의해서 공급된다. Arduino 보드에 공급된 전원은 Arduino 보드의 Vin핀과 5V핀을 이용하여 다시 모터 제어 모듈과 모터에 부착된 엔코더에 각각 전원을 공급한다. Arduino 보드 모터 제어 모듈의 Vin핀에서 모터 제어 명령에 따라 최종적으로 모터에 전원을 공급함으로써 모터를 동작시킨다.

그림 4. 2단부로 구성되어있는 로봇 (a) 로봇의 1단부 (b) 로봇의 2단부
Fig. 4. two part of robot structure (a) robot part 1 (b) robot part 2

2. 프로세서 성능에 따른 DP-SLAM의 결과물 측정

첫 번째 실험은 프로세서의 성능이 증가했을 경우, 즉 로봇에서 DP-SLAM의 Computational 오버헤드가 하드웨어적으로 개선되었을 경우에 구축된 결과맵이 더 정확해지는지 알아보기 위한 실험이다. 구체적인 수치로 평가하기 위해 PMC지수를 사용하였고, 로봇의 하드웨어를 업그레이드 하여 실험을 하였다. 기존의 로봇을 Bot, 하드웨어를 업그레이드

한 로봇은 Up-bot으로 칭한다.

표 3은 프로세서의 제원이며, 표 4에서는 로봇의 이동속도별 프로세서 성능 차이로 발생하는 결과맵의 완성도가 다르게 변하는 것을 PMC 지수로 확인할 수 있다.

로봇의 이동속도가 0.025m/s일 때, 결과맵 완성도는 3.9%정도의 완성도가 증가되며, 각 이동속도별로 완성도가 증가되는 것을 볼 수 있다. 로봇의 이동속도가 증가할 경우 맵의 완성도가 떨어지는 현상은 Computational 오버헤드를 처리하지 못한 원인이 되며, 프로세서의 성능 향상을 통해 개선할 수 있다는 것을 본 실험에서는 보여준다.

표 3. Bot과 Up-Bot의 프로세서 제원
Table 3. Bot and Up-Bot specification

Bot	CPU Intel Pentium M processor 1.5 GHz
	RAM 740MB
Up-bot	CPU Intel Core™ i-5 CPU M430 @ 2.27 GHz
	RAM 1GB

표 4. Bot과 Up-Bot의 PMC 차이
Table 4. Bot and Up-bot PMC comparison

로봇의 이동속도 (m/sec)	Bot PMC	Up-bot PMC
0.025	94.4%	98.3%
0.036	86.1%	98.4%
0.047	83.1%	96.1%
0.058	80.2%	95.8%
0.070	68.2%	87.6%
0.082	52.7%	70.8%

3. Idea Model : 로봇과 스마트폰의 자원공유를 통한 DP-SLAM의 구현

본 실험에서는 제안한 Idea Model, SLAM Task의 연산을 로봇이 스마트폰을 활용하여 Computational 오버헤드를 개선하는 아이디어를 구현하고 기존의 Basic Model과 비교하여, 성능을 평가하였다.

본 실험에서는 WIFI 환경에서 로봇과의 통신을 연결하였고, 통신으로 송수신되는 데이터가 단순하기 때문에 문제가 발생하지 않았다. 하지만 3G망이나 4G망에서는 패킷 로스로 인해 본 실험과 다른 결과가 나올 수 있다.

로봇과 스마트폰이 연결되어 자원공유가 시작되면, 로봇은 환경정보(레이저 센서데이터 181개)와 로봇의 위치정보(X, Y, Theta)를 스마트폰으로 전송하게 되며, 스마트폰은 SLAM Task로 연산하여 만들어진 맵과 로봇의 X, Y, Theta를 정보를 로봇으로 전송한다. 로봇에서는 저장되어있는 맵과 로봇의 위치, 방향정보를 업데이트하고 업데이트된 정보를 토대로 이동경로를 만들고 로봇은 움직이며 센서데이터를 전송하는 순환을 한다. 순환의 구조는 그림 5에서 볼 수 있다.

Idea Model 구현 결과는 로봇의 다양한 이동속도에서 결과맵을 PMC 지수로 평가하였다. 그림 6은 로봇의 이동속도별 Basic Model과 Idea Model의 성능을 cycle과 만들어진 맵의 결과물 형태로 보여준다. 1cycle은 SLAM의 모듈전체가 한바퀴 순회하는 것을 의미한다. 0.036m/sec의 이동속도의 Basic Model은 84회의 SLAM cycle을 통해서 맵을 완성하였고, PMC 지수는 86.1%이다. Idea Model의 경우 Basic Model보다 추가적인 6번의 연산, 즉 90회의 SLAM cycle을

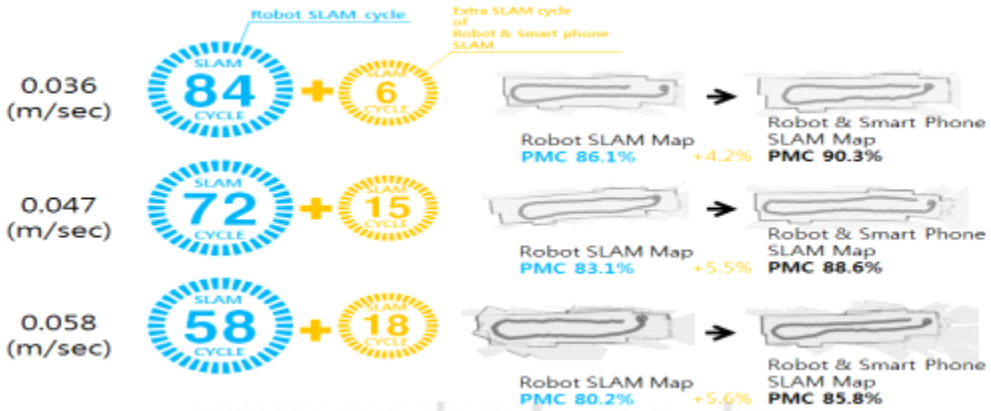


그림 5. 로봇의 각 이동속도(0.036, 0.047, 0.058)에서 Basic Model과 Idea Model의 결과 비교
Fig. 5. Basic model and Idea Model result comparison in Robot speed(0.036, 0.047, 0.058)

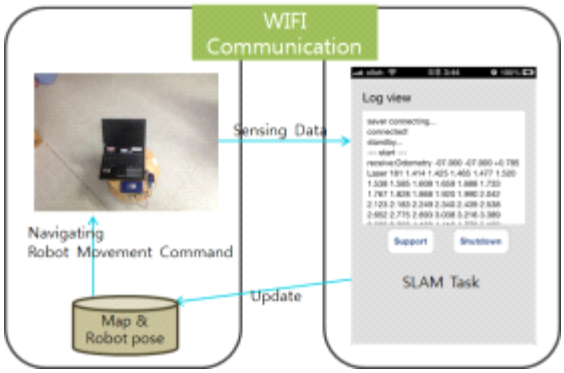


그림 5. 로봇과 스마트폰의 통신 구조도
Fig. 5. Architecture of communication between SmartPhone and Robot

통해 맵을 완성하였고, PMC지수는 90.3%로 Basic Model보다 4.2% 맵의 완성도가 개선되었다. 0.047m/sec의 경우 15cycle, 0.058m/sec의 경우 18cycle을 추가로 연산한 것을 확인 할 수 있다. 또한 그림 7에서는 Basic Model과 Idea Model의 성능을 PMC 수치로 평가한 것을 볼 수 있다.

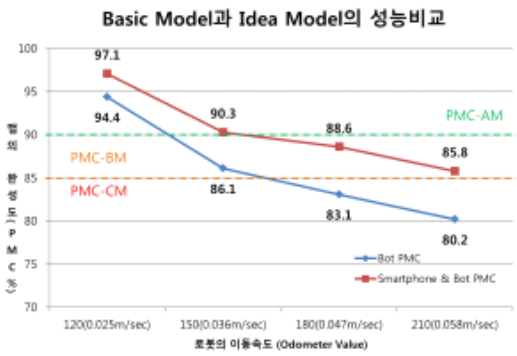


그림 7. Basic Model과 Idea Model의 PMC 비교
Fig. 7. Basic Model and Idea Model PMC comparison

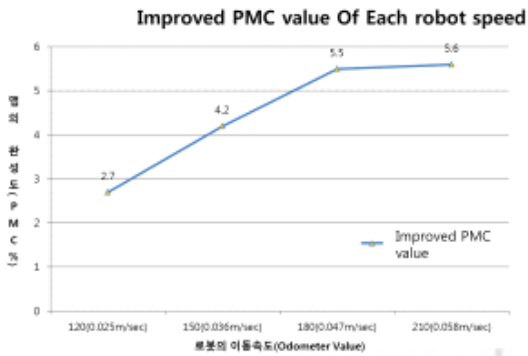


그림 8. Idea Model의 로봇의 이동속도별 PMC 증가수치
Fig. 8. Improved PMC value of each robot speed

동일한 속도에서 Idea Model의 경우가 더 많은 cycle횟수를 보이며 이것은 더 많은 연산을 처리했다는 것을 의미한다. 각각의 이동속도에서 맵이 개선되는 것을 확인할 수 있으며, 이동속도 0.047의 경우 15cycle, 0.058의 경우 18cycle로 연산되는 수치가 상승하는 것을 볼 수 있다. 이유로는 스마트폰과 로봇의 통신으로 인해 발생하는 통신오버헤드는 일정하게 발생하는데, 이 때 Idea Model로 얻는 연산적 이득이 점점 커지기 때문에 로봇의 속도가 증가할수록 일정수준까지 아이디어 모델의 효과가 증가하는 것을 그림 8에서 볼 수 있다. 로봇의 이동속도가 연산속도가 따라갈 수 없는 속도 이상(현 로봇 0.059m/sec이상)으로 증가한다면 추가적인 cycle횟수는 늘어나지만 무의미한 맵을 생산하였다.

로봇과 스마트폰의 자원공유를 통해서 맵의 완성도를 개선한 Idea Model의 실험 결과를 그림 7에서 그래프로 보여주며, 각각의 이동속도에서 PMC 등급이 한 등급씩 상향된 것을 볼 수 있다. Idea Model의 결과 맵 완성도가 개선되었다는 것은 스마트폰의 프로세싱 자원을 활용하여 Computational 오버헤드를 개선하였다는 것을 의미한다.

그림 7, 8에서 보이는 PMC 개선 수치는 로봇과 스마트폰의 성능에 따라서 상대적으로 변할 수 있다.

V. 결론

본 논문에서 제시한 Idea Model의 성능을 확인하기 위한 실험을 통해서 유효한 맵을 만들어낼 수 있는 로봇의 임계속도가 0.036m/sec에서 0.058m/sec로 증가한 것을 확인하였다. 이것은 유휴시간중인 스마트폰의 자원을 활용하여 SLAM의 Computational 오버헤드를 개선했다는 것을 의미한다. 또한 로봇의 이동속도가 증가할수록 스마트폰의 자원공유로 얻는 오버헤드 개선효과가 통신모듈로 인해 추가적으로 발생하는 오버헤드 부분보다 커지는 것을 확인 하였다.

향후 스마트폰의 성능과 통신 알고리즘의 성능이 향상 될수록 Idea Model의 SLAM Computational 오버헤드 개선효과가 증가할 것이라고 생각한다.

차후 본 Idea Model은 스마트폰 한 대가 아닌 여러 대의 스마트폰으로 자원공유를 할 수 있는 구조로 확장하여, SLAM의 Computational 오버헤드를 개선할 수 있도록 향후 연구를 진행할 것이다.

참고문헌

〔1〕 Hugh Durrant-Whyte and Tim Bailey, "Simultaneous Localization And Mapping: Part 1", IEEE Robotics & Automation Magazine, pp. 99-108, 2006.

〔2〕 G. Dissanayake, P. Newman, H.F. Durrant-Whyte, S. Clark, and M. Csobra, "A Solution to the Simultaneous Localization And Mapping (SLAM) Problem", IEEE Transactions of Robotics and Automation, vol. 17, no. 3, pp. 229-241, 2001.

〔3〕 Kurt-Yavuz Z, Yavuz S, "A comparison of EKF, UKF, FastSLAM2.0, and UKF-based FastSLAM algorithms", Intelligent Engineering Systems (INES) on IEEE 16th International Conference, 2012

〔4〕 Austin Eliazar, Ronald Parr, "DP-SLAM: Fast, Robust Simultaneous Localization and Mapping Without Predetermined Landmarks", International Joint Conferences on Artificial Intelligence, 2003.

〔5〕 Tim Bailey & Hugh Durrant-Whyte and, "Simultaneous Localization and Mapping (SLAM): Part II State of the Art", IEEE Robotics & Automation Magazine, 2006.

〔6〕 M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, ""FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges", International Joint Conferences on Artificial Intelligence, pp. 1151-1156, 2003.

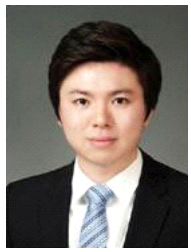
〔7〕 A. Eliazar and R. Parr, "DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks", International Joint Conferences on Artificial Intelligence, 2003.

〔8〕 Park Jeong-gyu, Jeon Heung-seok, No Sam-hyeok, "A Robot Coverage Algorithm Integrated with SLAM for Unknown Environments", KSCI , 2010.

〔9〕 Maffei.R, Jorge.V, Kolberg.M, "Segmented DP-SLAM", Intelligent Robots and Systems (IROS), 2013.

〔10〕 Wieser. I. Ruiz A.V., Frassl. M, "Autonomous robotic SLAM-based indoor navigation for high resolution sampling with complete coverage", Position, Location and Navigation Symposium - PLANS, 2013.

저 자 소 개



이 철 원

2012: 건국대학교 컴퓨터공학과 졸업
현 재: 건국대학교
컴퓨터정보과 박사과정
관심분야: 컴퓨터공학
Email : e10000won@gmail.com

전 흥 석

2002: 건국대학교 컴퓨터공학과 교수
현 재: 건국대학교 컴퓨터공학과 교수
관심분야: 컴퓨터공학
Email : hsjeon@kku.ac.kr