韓國컴퓨터情報學會 論文誌
第15卷 第1號, 2010. 1.

2010-15-1-1-7

미지의 환경에서 동작하는 SLAM 기반의 로봇 커버리지 알고리즘

박 정 규*, 전 흥 석**, 노 삼 혁***

A Robot Coverage Algorithm Integrated with SLAM for Unknown Environments

Jung Kyu Park*, Heung Seok Jeon**, Sam H. Noh***

요 약

로봇이 동작하는 환경을 완벽하게 커버리지 하기 위해서는 전체 환경 지도를 가지고 있어야 한다. 그러나 대부분의 기존 커버리지 알고리즘은 로봇이 동작하기 전 사전에 생성된 지도가 있어야 동작 한다. 이런 이유로 기존의 커버리지 알고리즘은 미지의 환경에 바로 적용할 수 없는 문제를 가지고 있다. 미지의 환경에서 로봇이 모든 영역을 커버리지 하기 위해서는 로봇 스스로 환경 지도를 생성할 수 있어야 한다. 본 논문에서는 SLAM 알고리즘을 통합하여 미지의 환경에서 로봇이 환경 지도를 생성하며 생성된 지도를 기반으로 커버리지를 수행하는 DmaxCoverage 알고리즘을 제안한다. 시뮬레이션 실험을 통해서 DmaxCoverage 알고리즘이 기존의 커버리지 알고리즘에 비해서 효율적임을 증명하였다.

Abstract

An autonomous robot must have a global workspace map in order to cover the complete workspace. However, most previous coverage algorithms assume that they have a grid workspace map that is to be covered before running the task. For this reason, most coverage algorithms can not be applied to complete coverage tasks in unknown environments. An autonomous robot has to build a workspace map by itself for complete coverage in unknown environments. Thus, we propose a new DmaxCoverage algorithm that allows a robot to carry out a complete coverage task in unknown environments. This algorithm integrates a SLAM algorithm for simultaneous workspace map building. Experimentally, we verify that DmaxCoverage algorithm is more efficient than previous algorithms.

▸ Keyword : Coverage algorithm, Mobile robot, SLAM

• 제1저자 : 박정규 교신저자 : 전흥석
• 투고일 : 2009. 11. 26, 심사일 : 2009. 12. 09, 게재확정일 : 2010. 01. 26.
* 홍익대학교 컴퓨터공학과 박사과정 ** 건국대학교 컴퓨터응용과학부 부교수 *** 홍익대학교 정보컴퓨터공학부 교수
※이 논문은 2006년도 건국대학교 학술진흥연구비 지원에 의한 논문임

Ⅰ. 서 론

서비스 로봇이 새로운 환경에서 주어진 임무를 수행하기 위해서는 정확한 이동이 필요하다. 로봇이 정확한 이동을 하기 위해서는 이동 경로를 결정하는 경로 탐색(Navigation) 알고리즘이 요구된다. 많은 경로 탐색 알고리즘에 기본이 되는 것으로 커버리지(Coverage) 알고리즘을 들 수 있다[1][2][3][4][5]. 커버리지 알고리즘은 로봇이 동작하는 환경의 모든 영역을 방문할 수 있도록 이동 경로를 생성해 주는 알고리즘이다.

기존의 많은 커버리지 알고리즘들은 구현의 간편함 때문에 환경 내에서 무작위로 이동을 하는 랜덤 기반 알고리즘을 사용하고 있다[6]. 그러나 랜덤 알고리즘을 기반으로 동작하는 로봇은 환경을 고려하지 않고 무작위로 이동을 하기 때문에 동작하는 영역을 모두 커버리지 할 수 없거나, 모두 커버리지 하기 위해서는 많이 시간이 든다는 한계를 가지고 있다.

로봇이 동작하는 환경을 모두 커버리지 못하는 랜덤 알고리즘의 문제를 해결하기 위해 완벽한 커버리지를 고려하는 많은 연구가 진행되었다[7][8][9][10][11]. 진행된 커버리지 연구들은 로봇이 동작하기 전에 환경에 대한 정보를 가지고 있는 것을 가정하고 있다. 로봇이 환경 전체에 대한 사전 정보를 가지고 있고 시작 위치와 종료 위치를 알고 있을 때 모든 영역을 커버리지할 수 있는 이동 경로를 생성할 수 있다. 그러나 로봇이 사전에 환경에 대한 정보를 가지고 있지 않을 경우에는 기존 연구의 알고리즘을 바로 적용할 수 없는 문제점을 가지고 있다.

로봇이 사전 정보를 가지고 있지 않은 상태에서 모든 영역을 커버리지 하기 위해서는 로봇 스스로 환경에 대한 정보를 수집할 수 있어야 한다. 환경 정보로 가장 많이 사용되는 환경 지도를 로봇이 자체적으로 생성할 수 있다면, 생성하는 지도를 기반으로 커버리지 알고리즘을 적용하여 모든 영역을 커버리지 할 수 있다. 이렇게 로봇이 자체적으로 환경 지도를 생성하기 위해서는 Simultaneous Localization and Mapping (SLAM) 문제를 해결해야 한다[12][13][14][15][16][17]. SLAM 문제는 로봇이 사전 정보가 없는 상태에서 로봇 스스로 환경 지도를 생성하며, 생성된 지도를 기반으로 로봇의 위치를 추정하는 것을 의미한다.

본 논문에서는 환경의 사전 정보를 가지고 있지 않은 상태에서 영역을 커버리지 할 수 있는 DmaxCoverage 알고리즘을 제안한다. DmaxCoverage 알고리즘은 로봇이 처리해야 하는 영역을 작은 영역으로 나누어 처리하는 커버리지 알고리즘과 환경 지도를 생성하는 SLAM 알고리즘으로 구성되어 있다. SLAM 알고리즘을 통합한 DmaxCoverage 알고리즘은 사전 정보가 없는 상황에서 로봇이 동작할 수 있도록 로봇 스스로 생성한 지도를 커버리지에 사용한다. 실험 결과에 따르면 Dmax Coverage 알고리즘은 환경 정보가 없는 상태에 적용할 수 있으며 기존의 커버리지 알고리즘에 비해 최대 38% 효율적임을 알 수 있다.

논문은 다음과 같이 구성되어 있다. 다음 2장 에서는 논문의 작성의 동기가 되는 관련 연구를 설명한다. 3장에서는 본 논문에서 제안하는 DmaxCoverage 알고리즘에 대해서 설명하고, 4장에서는 실험과 그 결과에 대해서 설명한다. 마지막 5장에서는 결론과 향후 계획에 대해서 설명한다.

Ⅱ. 관련 연구

본 논문은 로봇 커버리지 알고리즘과 환경 정보 생성을 동시에 고려하고 있다. 기존 커버리지 알고리즘 분야에서 많은 연구들이 이루어 졌다. 대표적인 연구로 바둑판식 경로 알고리즘(Boustrophedon Path algorithm)을 들 수 있다[8]. 바둑판식 경로 알고리즘은 Zelinsky의 연구를 확장한 것으로 현재 판매되는 청소 로봇에서 많이 사용되고 있는 알고리즘이다[9][10]. 바둑판식 알고리즘은 커버리지 하는 영역이 작거나 영역 내부에 장애물이 없을 경우 좋은 효율을 보여준다. 그러나 로봇이 동작하는 환경이 복잡 할 때 처리한 영역을 반복해서 이동하는 문제로 성능이 떨어지는 문제를 가지고 있다.

커버리지 알고리즘의 기반이 되는 연구로 Distance Transform을 들 수 있다[7]. Distance Transform은 그리드 기반의 지도에서 두 셀 사이의 최소 거리만을 구해 경로를 생성한다. 최소 거리를 기반으로 생성된 경로는 로봇이 지나갈 수 없는 물체 바로 옆의 경로나 아주 좁은 경로를 지날 수 있기 때문에 실제 로봇에 적용하는데 문제가 있다. 또한 알고리즘을 사용하기 위해서는 사전에 생성된 그리드 환경 지도가 필요하다. 이러한 문제를 가지고 있지만 이 연구를 기반으로 많은 커버리지 연구가 이루어 졌다.

앞의 문제를 해결하기 위해 Path Transform이 제안되었다[9][10]. Path Transform은 로봇의 안전한 이동 경로를 생성하기 위해 두 셀 사이의 최소 거리와 로봇과 물체의 거리를 고려하고 있다. 로봇의 이동 경로를 결정할 때 좁은 경로를 생성하지 않도록 물체와 거리가 먼 곳의 셀에 가중치를 두어 경로 선택에 사용하고 있다. 그러나 Path Transform도 완벽한 커버리지를 사용하기 위해서 사전에 생성된 그리드 기반의 지도를 가지고 있어야 한다. 또한 로봇이 동작하는 환경이 복잡할수록 로봇의 회전수가 많아져 전체 수행시간이 길어지는 문

제점을 가지고 있다.

Triangular-Cell-Based 지도 알고리즘은 Distance Transform을 확장한 방법이다[11]. 이 알고리즘은 완벽한 커버리지 경로를 구성하기 위해 로봇이 처음 동작할 때 전체 환경 지도를 생성하는 것을 가정하고 있다. 초기 로봇이 환경의 가장 외곽 벽을 따라 움직이며(Wall-Following) 환경을 파악 하면서 지도를 생성하고, 생성한 지도를 기반으로 완벽한 커버리지 경로를 생성하게 된다. 그러나 로봇의 자원이 한정되어 있거나, 환경이 닫힌 상태가 아니라면 지도를 생성할 수 없고, 커버리지 경로를 생성할 수 없어 전체 영역을 처리할 수 없다.

커버리지 연구가 많이 이루어 졌으나 로봇이 환경에 대한 정보가 없이 동작하기 위해서는 기존의 커버리지 연구를 바로 활용할 수 없는 문제를 가지고 있다.

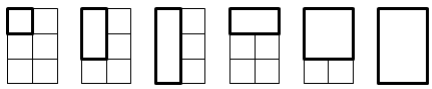
로봇이 사전 정보가 없는 상태에서 지도를 생성하며 위치를 추정하는 SLAM 문제를 해결하는 SLAM 알고리즘에 대한 연구가 많이 이루어졌다[12][13][14][15][16][17]. SLAM 문제를 해결하는 대표적인 알고리즘으로 Extended Kalman Filter (EKF)-SLAM과 FastSLAM을 예로 들 수 있다[16][17]. EKF-SLAM은 로봇 연구 분야에서 가장 많이 사용된 알고리즘이다. 그러나 EKF-SLAM은 로봇이 이동하면서 환경 내에 존재하는 위치 표식(Landmark)을 발견할 때마다 로봇의 위치 및 환경의 위치 정보를 반복적으로 업데이트하게 된다. 이때 새로 업데이트되는 정보와 기존에 가지고 있는 정보 모두를 업데이트하기 때문에 많은 계산이 필요하다는 단점을 가지고 있다.

EKF-SLAM이 가지고 있는 업데이트의 많은 계산량과 복잡도 문제를 해결하기 위해서 FastSLAM 알고리즘이 제안되었다[17]. FastSLAM 알고리즘에서는 위치를 업데이트하는 계산 복잡도 문제를 줄이기 위해서 로봇의 위치 추정과 위치 표식 추정을 나누어 처리하여 위치 표식을 추정할 때 기존에 가지고 있는 로봇의 모든 위치 정보를 다시 계산하지 않는다. 이 결과 업데이트되는 계산량과 복잡도를 줄여 기존의 SLAM 알고리즘에 비해서 빠른 처리가 가능하다.

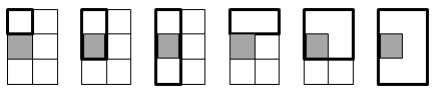
본 논문에서는 SLAM 알고리즘을 커버리지 알고리즘과 통합하여 로봇이 사전 정보가 없는 상황에서 동작할 수 있는 방안을 제안하고 있다.

III. DmaxCoverage 알고리즘

DmaxCoverage 알고리즘은 로봇이 동작하는 환경에 대한 정보가 없는 상황에서 모든 영역을 커버할 수 있는 커버리지 알고리즘이다. 로봇이 동작하는 모든 영역을 커버하기 위해서는 환경에 대한 정보를 알고 있어야만 한다. DmaxCoverage



(a) 물체를 고려하지 않은 사각형 타일링 예
(a) Example of Rectangle Tiling without obstacles



(b) 물체를 고려하는 사각형 타일링 예
(b) Example of Rectangle Tiling with an obstacles

그림 1. 사각형 타일링 예
Fig. 1. Examples of Rectangle Tiling

알고리즘이 환경 정보가 없는 상태에서 커버리지 임무를 수행하기 위해서는 로봇이 실시간으로 환경 지도를 생성할 수 있어야 한다. DmaxCoverage 알고리즘은 실시간 지도를 생성할 수 있도록 Simultaneous Localization and Mapping(SLAM) 알고리즘을 통합 하였다.

3.1 DmaxCoverage 알고리즘

DmaxCoverage 알고리즘의 가장 큰 특징은 로봇이 동작하는 환경을 효율적으로 처리할 수 있는 영역으로 나누어 처리하는 것이다. 바둑판식 커버리지 알고리즘을 기반으로 하는 DmaxCoverage 알고리즘은 바둑판식 알고리즘의 효율성을 최대화하기 위해 로봇이 처리할 수 있는 영역을 사각형 형태로 나누어 처리하게 된다. 영역을 사각형 형태로 나눌 때 바둑판식 알고리즘의 효율을 최대화하기 위해서 각 사각형은 장애물을 포함하지 않도록 나누어야한다. 이렇게 나누어진 사각형은 장애물을 포함하지 않기 때문에 바둑판식 알고리즘으로 최대의 효과를 낼 수 있다.

위에 설명한 것과 같이 DmaxCoverage 알고리즘은 처리해야 하는 영역을 나누기 위해 사각형을 나누는 작업을 하게 된다. 영역을 사각형 형태로 나누기 위해 사각형 타일링 방법(Rectangle Tiling Scheme)을 사용하였다[18].

그림 1은 사각형 타일링의 예를 보여주고 있다. 그림 1(a)의 가장 왼쪽에 있는 6개의 셀로 구성된 그리드 형태의 사각형을 타일링을 한다는 것은 사각형 내에 존재하는 작은 사각형을 모두 구하는 것과 같다. 예를 들어 가장 작은 단위의 사

각형은 하나의 셀로 총 6개, 세로 형태의 2셀로 구성된 사각형은 총 4개, 이런 방식으로 사각형을 타일링 했을 경우 총 18개의 사각형으로 나눌 수 있다.

그림 1(b)는 그림 1(a)와 다르게 그리드 사각형 내에 물체를 나타내는 셀(회색으로 표시)이 존재하는 경우를 보여준다. 물체가 존재하는 그리드 사각형을 타일링 할 때는 나누어진 작은 사각형에 물체가 포함되어 있을 경우 타일링된 목록에서 제거한다. 예를 들어 가장 작은 단위의 사각형으로 나누면 물체를 포함하는 사각형을 제외하여 총 5개의 사각형으로 나눌 수 있다. 이런 방식으로 나머지 사각형을 구하면 총 10개의 사각형을 구할 수 있다. 본 논문에서는 사각형 타일링한 목록을 구성할 때 드는 시간과 생성된 목록에서 모든 영역을 처리하는 영역을 선택할 때의 시간을 줄이기 위해 물체를 고려하는 타일링을 수행하고 있다.

사각형 타일링 방법은 수학적인 방법으로 사각형 안에 가능한 모든 사각형을 찾는 방법이다. $m \times n$ 사각형은 $N(m,n)$의 사각형들로 나눌 수 있다. 예를 들어 사각형의 왼쪽 아래 좌표를 (i,j)라 하면 오른쪽 위쪽 좌표는 $(m-i)(n-j)$개가 될 수 있다. 이렇게 만들어 지는 사각형의 개수는 수식 (1)로 계산 될 수 있다.

$$N(m,n) = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(m-i)(m-j) = \frac{1}{4}m(m+1)n(n+1) \quad (1)$$

이제부터 DMaxCoverage 알고리즘의 실행 과정을 설명하기 위해 그림 2를 활용하기로 한다. 로봇이 동작해야 하는 환경에 최초 투입되면 로봇은 처음 처리해야 하는 영역을 처리하기 위해 로봇이 가지고 있는 센서를 사용하여 환경 지도를 작성하게 된다. MinimumBoundingRectangle() 함수에서는 생성된 환경 지도를 기반으로 로봇이 처리할 수 있는 최소의 사각형(Minimum Bounding Rectangle, MBR)을 선택하게 된다(그림 2: 라인 4). 로봇이 동작하는 도중에 환경 지도가 업데이트 되어 로봇이 처리할 수 있는 영역이 변하게 되면 MBR 영역 또한 변하게 된다.

초기 지도가 생성된 후 MBR 영역이 결정되면 RectangleTilingwithoutObstacle(mbr) 함수에서는 선택된 MBR을 기준으로 사각형 타일링을 실행하여 처리 가능한 사각형들로 분리한다(그림 2: 라인 5). 이때 MBR 내부에는 장애물 영역과 처리 가능한 빈 영역이 모두 포함 되어있기 때문에 사각형 타일링을 할 때는 그림 1(b)와 같이 장애물이 포함된 사각형을 제외하게 된다. 장애물을 제외한 사각형 타일링을 실행함으로 생성되는 사각형 목록의 처리 시간을 줄일 수 있다.

```
Algorithm: DmaxCoverage(deadline)
1: Global Map
2: Bool MBRchanged, RECchanged
3: MBRchanged := FALSE, RECchanged := FALSE
4: mbr := call MinimumBoundingRectangle()
5: rectangles := call RectangleTilingwithoutObstacle(mbr)
6: setofrectangles := call SetCovering(rectangles)
7: r1 := call getR1(setofrectangles,deadline)
8: while(r1 is not null and deadline is not exhausted)
9:       move to the initial position of r1
10:      call CoverR1(r1)
11:      if(MBRchanged==True) Then
12:          MBRchanged := False
13:          goto line 4
14:      endif
15:      if(RECchanged==True) Then
16:          RECchanged := False
17:          goto line 5
18:      endif
19:      r1 := call getR1(setofrectangles)
20: endofwhile
END DmaxCoverage
```

그림 2. DmaxCoverage 알고리즘
Fig. 2. The DMaxCoverage algorithm

본 알고리즘에서는 사각형 타일링에 의해 생성된 사각형의 목록중에서 바둑판식 알고리즘으로 처리 했을 때 가장 효율이 큰 사각형의 순서로 선택하여 처리를 하게 된다. 그러나 이 사각형을 선택하는 것은 잘 알려진 Set Cover 문제로 NP-Complete 한 것으로 알려져 있다[19]. 즉 생성된 사각형의 개수가 증가함에 따라 사각형을 선택하는 시간이 지수적으로(exponential) 커지는 문제가 발생한다. 이런 이유로 본 알고리즘에서는 타일링된 사각형 목록에서 사각형이 겹치지 않는 범위에서 가장 큰 순서대로 선택하는 그리디(greedy) 방법으로 사각형 선택 문제를 해결하고 있다.

사각형 타일링에서 생성된 사각형 목록을 사용하여 SetCovering(rectangles) 함수를 수행하면 로봇이 처리할 수 있는 영역으로 구성된 Set Cover 목록이 생성된다(그림 2: 라인 6). 로봇은 자신이 처리할 수 있는 동안 목록의 순서대로 이동하면서 영역을 처리하게 된다. GetR1(rectangles) 함수를 이용하여 Set Cover 목록에서 로봇이 가지고 있는 자원(예, 베터리)으로 처리할 수 있는 최대의 영역을 선택 한 후(그림 2: 라인 7),

CoverR1(r1) 함수를 통해 바둑판식 알고리즘으로 영역을 처리하게 하게 된다(그림 2: 라인 10). 이 과정을 Set Cover 목록이 다 처리되거나, 로봇이 가지고 있는 자원을 다 소모할 때 까지 반복한다.

로봇이 영역을 처리하는 동안 2가지 상황이 발생할 수 있다. 첫 번째로 로봇이 생성하는 환경 지도가 현재 MBR의 크기보다 확장될 수 있다. 이때는 새로운 MBR을 구하고, 사각형 타일링, Set Cover를 다시 수행하여 새로 생성된 사각형 목록을 기반으로 임무를 수행하게 된다(그림 2: 라인 11). 두 번째로 MBR 내부 물체의 이동 등으로 인해 내부 구조가 변경될 수 있다. 이 경우에는 사각형 타일링과 Set Cover를 수행하여 새로운 사각형 목록을 구성한 후 목록을 기반으로 커버리지를 수행하게 된다(그림 2: 라인 15).

3.2 SLAM 알고리즘

DmaxCoverage 알고리즘에서는 환경 지도를 생성하기 위해 SLAM 알고리즘을 사용하고 있다. 사용한 SLAM 알고리즘은 파티클 필터 기반의 Adaptive Monte Carlo Localization (AMCL) 방법으로 로봇의 위치를 추정할 때 사용한다[13][14]. AMCL은 주어진 환경 지도에서 로봇의 위치를 추정할 때 사용된다. 본 알고리즘에서는 AMCL 알고리즘에서 사용할 환경 지도를 작성하기 위해 로봇이 위치한 곳에서 360도 회전을 하면서 최초 지도를 생성한다. 최초 생성된 지도를 기반으로 DmaxCoverage 알고리즘을 적용하여 영역의 커버를 시도하는 동시에 AMCL 방법을 사용하여 로봇의 위치를 추정한다. 또한 로봇이 영역을 커버하면서 센서를 통해 수집한 정보를 사용하여 환경 지도를 계속 업데이트 하게 된다.

IV. 실험 및 결과

본 논문에서 제안하는 알고리즘의 효율성과 성능을 증명하기 위해서 시뮬레이션 실험을 수행하였다. 실험에서는 랜덤 알고리즘과 바둑판식 알고리즘을 비교 대상으로 하였다.

4.1 실험 환경

논문에서 제안한 알고리즘과 기존의 알고리즘을 비교 평가하기 위해서 Player/Stage 시뮬레이션 환경을 사용하였다[20]. 실험에 사용된 시뮬레이션 프로그램의 버전은 Player 2.1.2/Stage 3.0.1이고, SICK 레이저(LMS200)을 장착한 Pioneer 2DX 로봇을 사용하였다. 로봇은 주행 속도는 0.4m/s, 회전 속도는 0.2 m/s로 설정하여 사용하였다.

로봇이 동작하는 환경으로 그림 3과 같이 아파트 형태의 주거지를 모델링한 지도를 사용하였다. 그림 3의 지도에서 표시된 P1, P2, P3은 로봇의 초기 시작 위치를 나타낸다.

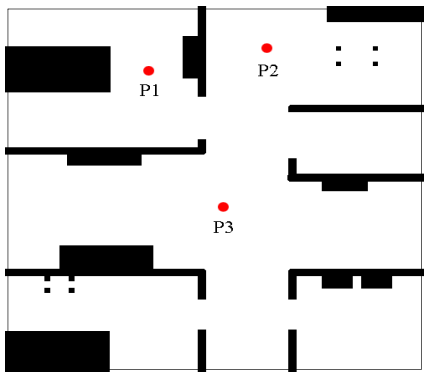


그림 3. 환경 지도와 3곳의 초기 시작 위치 (P1, P2, P3)
Fig. 3. Map and the three initial start positions (P1, P2, P3) of the robot for the experiment

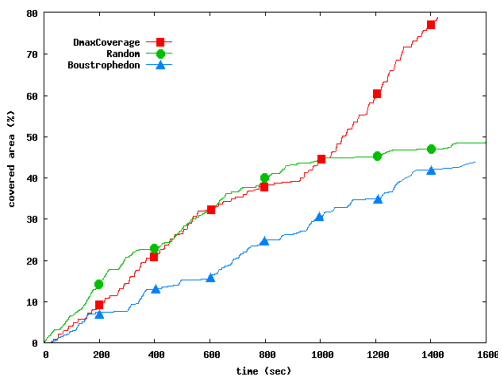
4.2 실험 결과

4.2.1 시간에 따른 커버리지 효율 변화

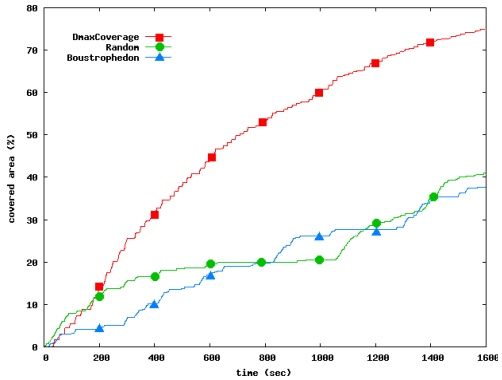
DmaxCoverage 알고리즘과 랜덤, 바둑판식 알고리즘의 성능 평가를 하기 위해 동일 환경에서 각각의 알고리즘을 수행하여 커버리지 효율을 비교해 보았다. 각 실험은 그림 3의 환경에서 수행되었으며 로봇의 초기 시작 위치를 다르게 조정하였다. 로봇의 시작 위치는 그림 3에서 P1, P2, P3로 표시하였다.

그림 4는 DmaxCoverage 알고리즘과 랜덤, 바둑판식 알고리즘을 사용했을 때 로봇의 동작 시간에 따른 커버리지 완료 비율을 보여주고 있다. 실험 결과에 따르면 DmaxCoverage 알고리즘이 랜덤, 바둑판식 알고리즘에 비해서 최대 38% (P1 지점의 결과), 최소 21%의 성능 향상을 보였다(P3 지점의 결과). DmaxCoverage 알고리즘은 로봇이 처리할 수 있는 영역을 작은 공간으로 나누어 처리함으로 영역의 커버리지 효율이 계속 증가하지만 랜덤, 바둑판식 알고리즘은 복잡한 구역에서 더 이상 청소를 하지 못하고 청소한 구역만을 반복해서 청소하는 문제가 발생하여 영역의 커버리지 비율이 증가하지 못하고 일정 시간 동안 정체되는 현상이 발생되고 있다.

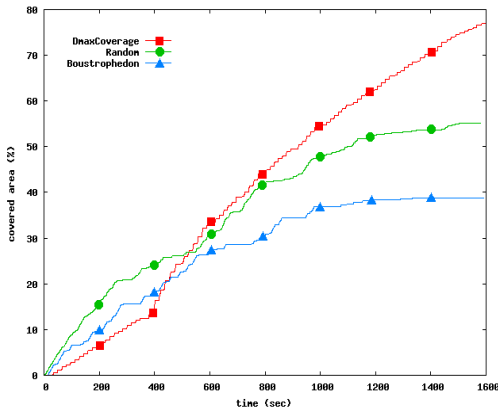
실험 결과 초기 5분 정도까지 DmaxCoverage 알고리즘의 성능이 랜덤 또는 바둑판식 알고리즘에 비해서 성능이 떨어지는 것을 알 수 있다. 이는 DmaxCoverage 알고리즘이 초기 환경지도를 생성할 때 사용하는 시간이 크기 때문이다. 로봇이 초기 환경 지도를 생성하기 위해 시작 위치에서 360도 회전에 사용하는 시간이 약30초, 초기 타일링 시간으로 약 1초 정도 사용을 하고 있다. DmaxCoverage 알고리즘이 초기 환경지도



(a) P1 지점의 실험 결과
(a) Results from initial position P1



(b) P2 지점의 실험 결과
(b) Results from initial position P2



(c) P3 지점의 실험 결과
(c) Results from initial position P3

그림 4. 커버리지 완료 비율
Fig. 4. Covering completion rate

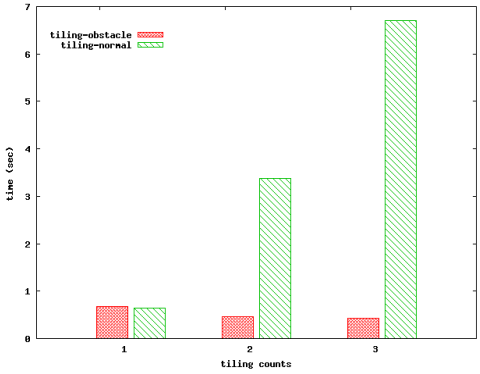


그림 5. 사각형 타일링 시간
Fig. 5. Rectangle tiling time

를 생성하기 위해 사용하는 시간 동안 다른 알고리즘은 바로 커버리지를 수행하여 그 커버리지 비율의 차이가 약 5분 정도까지 영향을 주고 있다. 그 이후에는 DmaxCoverage 알고리즘이 나머지 영역을 계속 커버하는 반면, 랜덤, 바둑판식 알고리즘을 사용하는 로봇은 커버한 영역을 반복 주행하는 문제로 커버리지 효율이 반전되는 것을 알 수 있다.

4.2.2 사각형 타일링 오버헤드

DmaxCoverage 알고리즘에서는 처리해야 하는 영역을 효율적으로 나누기 위해 사각형 타일링 방법을 도입하고 있다. 사각형 타일링 방법은 나누어야 하는 영역이 커짐에 따라 사각형 타일링 수행 시간이 비례해서 커지게 되는 문제를 가지고 있다. 제안하는 알고리즘에서는 사각형 타일링 수행 시간을 줄이기 위해서 두 가지를 고려하고 있다.

첫 번째로 MBR 개념을 사용하여 시간을 단축하고 있다. 예를 들어 32×32 셀의 그리드 형태의 영역이 있을 때 이 영역을 전체 형태로 보고 사각형 타일링을 수행하면 약 27만 개의 사각형 목록이 생성된다. 그러나 DmaxCoverage 알고리즘에서는 MBR 영역만을 사각형 타일링하여 초기 타일링 수행시 약 1만개의 사각형만을 생성하게 되어 타일링 목록 생성 시간과 Set Cover 시간을 단축하고 있다.

두 번째로 물체를 고려하는 타일링을 수행하고 있다. 로봇이 영역을 처리하면서 MBR 영역이 커지면 그 영역의 크기에 따라 사각형 타일링 시간이 커지게 된다. 그러나 DmaxCoverage 알고리즘에서는 이미 커버리지한 공간을 물체로 처리하여 사각형 타일링 시간을 줄이고 있다.

그림 5는 DmaxCoverage 알고리즘이 수행되면서 3번의 사각형 타일링이 수행되는 시간을 측정한 것이다. 사각형 타일링을 수행할 때 로봇이 처리한 영역을 물체로 처리하여 사각형 타일링에서 제외할 때의 타일링 시간(tiling-obstacle)과 처

리한 영역을 사각형 타일링에서 제외하지 않았을 때의 타일링 시간(tiling-normal)을 측정한 것이다. 사각형 타일링을 수행할 때는 로봇이 생성한 환경 지도의 해상도를 낮추어 32×32 셀의 그리드 지도를 만들어 사용하였다.

로봇이 초기 위치에서 환경 지도를 생성한 후 첫 번째 사각형 타일링을 실시하게 된다. DmaxCoverage 알고리즘이 처음 타일링을 실시할 때 총 면적에서 타일링 가능한 면적의 비율은 약 39%이고, 약 0.6초의 시간이 걸렸다. 3번 째 타일링이 수행되었을 때는 MBR이 확장되어 타일링 가능한 면적이 약 51%로 증가 하였으나 시간은 약 0.4초로 감소하였다.

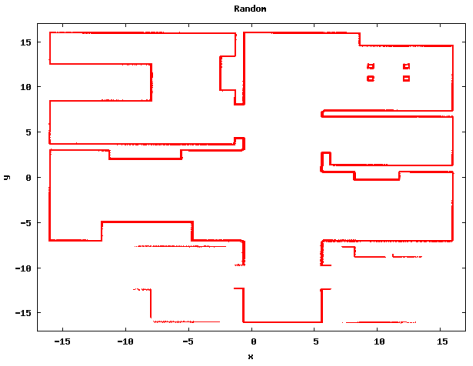
이와 반대로 로봇이 처리한 영역을 타일링에서 제외하지 않았을 경우에는 MBR이 커짐에 따라 타일링 시간이 커짐을 알 수 있다. 로봇이 처리한 영역을 타일링에서 제외하지 않는 타일링에 DmaxCoverage 알고리즘에서 3번 째 타일링을 수행할 때의 MBR 크기를 적용하여 타일링을 했을 경우 로봇이 처리한 영역을 물체로 처리하는 타일링에 비해서 6초 이상의 시간이 걸렸다.

4.2.3 환경 지도 생성

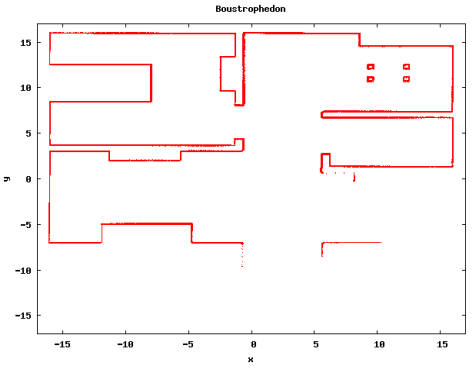
기존의 커버리지 알고리즘의 단점을 개선하기 위해 DmaxCoverage 알고리즘은 실시간으로 환경 지도를 생성하는 SLAM 알고리즘을 도입하였다. SLAM 알고리즘의 도입으로 DmaxCoverage 알고리즘에서는 사전에 환경 정보가 없는 상태에서 커버리지를 효율적으로 수행할 수 있다. 그림 6은 그림 2의 로봇 동작 환경(P3)에서 각 알고리즘을 30분 수행했을 때 생성한 환경지도를 보여주고 있다.

랜덤 알고리즘으로 생성한 환경 지도는 바둑판식 알고리즘 환경지도에 비해서 동작하는 환경을 좀 더 자세히 반영하고 있다. 실험에서는 바둑판식 알고리즘을 적용한 로봇이 환경의 위쪽 부분을 반복해서 처리하여 환경의 아래쪽 모습을 반영하지 못한 것을 알 수 있었다.

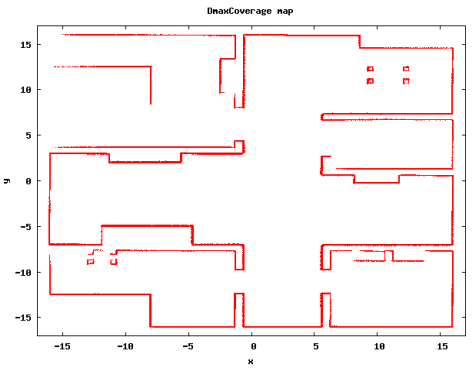
DmaxCoverage 알고리즘으로 생성한 환경지도 그림 6(c)는 그림 4의 실험 초기 DmaxCoverage 보다 더 좋은 결과를 보였던 랜덤 알고리즘의 환경지도 그림 6(a)에 비해서 환경을 잘 표현하고 있다. 랜덤 알고리즘의 지도가 환경을 잘 표현하기는 하지만 이미 처리된 영역을 반복 처리하는 문제로 인해 환경의 아래쪽 부분을 반영하지 못하고 있다. 그러나 DmaxCoverage 알고리즘은 로봇이 처리할 수 있는 영역으로 나누어 전체 영역을 처리하기 때문에 다른 알고리즘에 비해서 환경을 잘 표현하는 지도를 작성할 수 있었다.



(a) 랜덤 알고리즘으로 생성한 환경 지도
(a) Map built using Random algorithm



(b) 바둑판식 알고리즘으로 생성한 환경 지도
(b) Map built using Boustrophedon Path algorithm



(c) DmaxCoverage 알고리즘으로 생성한 환경 지도
(c) Map built using DMaxCoverage algorithm

그림 6. 커버리를 30분 수행 후 생성한 환경 지도
Fig. 6. Maps built after 30 minutes of coverage

V. 결론

본 논문에서는 새로운 커버리지 알고리즘으로 DmaxCoverage 알고리즘을 제안하고 있다. DmaxCoverage 알고리즘은 기존의 커버리지 알고리즘의 문제를 해결하기 위해 SLAM 알고리즘을 통합하고 있다. SLAM 알고리즘이 통합된 DmaxCoverage 알고리즘의 실험 결과를 통해서 DmaxCoverage 알고리즘이 기존 알고리즘이 처리 할 수 없었던 환경에서 동작할 수 있으며, 랜덤 알고리즘에 비해서 최대 38% 효율적임을 알 수 있었다.

현재 DmaxCoverage 알고리즘은 로봇이 동작하는 환경을 정적이라고 가정하고 있다. 이 문제를 해결하기 위해서 차후 연구과제로 동적인 환경에서 DmaxCoverage 알고리즘을 적용하는 방법에 대해서 연구할 예정이다. 또한 외부의 센서를 사용하는 환경에서 DmaxCoverage 알고리즘 효율을 높일 수 있는 방법에 대해서 연구할 예정이다.

참고문헌

〔1〕 R. D. Schraft, M. Hägele, and H. Volz, "Service robots: the appropriate level of automation and the role of users/operators in the task execution," In Proceedings of the International Conference Systems, Man, and Cybernetics, Vol. 4, pp.163-169, Le Touquet, France, Oct. 1993.

〔2〕 R. N. Carvalho, H. A. Vidal, P. Vieira and M. I. Ribeiro, "Complete Coverage Path Planning and Guidance for Cleaning Robots," In Proceedings of the IEEE International Symposium on Industrial Electronics, pp. 677-682, Guimaraes, Portugal, Jul. 1997.

〔3〕 S. H. Yoon, S. H. Park, B. J. Choi, and Y. J. Lee, "Path Planning for Cleaning Robots: A Graph Model Approach," In Proceedings of the International Conference on Control, Automation and Systems, pp. 2861-2864, Cheju, Korea, Oct. 2001.

〔4〕 S. C. Wong and B. A. MacDonald, "A topological coverage algorithm for mobile robots," In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol. 2, pp. 1685-1690, Las Vegas, U.S.A., Oct. 2003.

〔5〕 N. Agmon, M. Hozon, and G. A. Kaminka, "Constructing Spanning Trees for Efficient Multi-Robot Coverage," In Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1698-1703, Florida, U.S.A., May 2006

〔6〕 J. Jones, "Robots at the Tipping Point: The Road to the iRobot Roomba," IEEE Robotics and Automation Magazine, 2006.

〔7〕 R. A. Jarvis and J. C. Byrne, "Robot Navigation: Touching, Seeing and Knowing," In Proceedings of the Australian Conference on Artificial Intelligence, Nov. 1986.

〔8〕 H. Choset and P. Pignon, "Coverage Path Planning: the Boustrophedon Cellular Decomposition," In Proceedings of the International Conference on Field and Service Robotics, Canberra, Australia, Dec. 1997.

〔9〕 A. Zelinsky, R. A. Jarvis, J. C. Byrne, and S. Yuta, "Planning Paths of Complete Coverage of an Unstructured Environment by a Mobile Robot," In Proceedings of the International Conference on Advanced Robotics, pp. 533-538, Tokyo, Japan, Nov. 1993.

〔10〕 A. Zelinsky, "Using path transforms to guide the search for findpath in 2D," International Journal of Robotics Research, Vol. 13(4), pp. 315-325, Aug. 1994.

〔11〕 J. S. Oh, Y. H. Choi, J. B. Park, and Y. F. Zheng, "Complete Coverage Navigation of Cleaning Robots Using Triangular-Cell-Based Map," IEEE Transactions on Industrial Electronics, Vol. 51(3), pp. 718-726, Jun. 2004.

〔12〕 S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte. "A New Approach fo r Filtering Nonlinear Systems," In Proceedings of the American Control Conference, Vol. 3, pp. 1628-1632, Seattle, WA, Jun. 1995.

〔13〕 D. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello, "Bayesian Filtering for Location Estimation," IEEE Pervasive Computing, Vol. 2, No. 3, pp. 24- 33, Jul.-Sep. 2003

〔14〕 D. Fox, W. Burgard, and S. Thrun, "Active Markov Localization for Mobile Robots," Robotics and Autonomous Systems, Vol. 25, No. 3-4, pp. 195-207, Nov. 1998

〔15〕 S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust Monte Carlo Localization for Mobile Robots," In

Proceedings of the National Conference on Artificial Intelligence, Vol. 128, No. 1-2, pp. 99-141, May, 2000.
〔16〕 G. Dissanayake, P. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csobra, "A Solution to the Simultaneous Localisation and Mapping (SLAM) Problem," IEEE Transactions on Robotics and Automation, Vol. 17, pp. 229-241, 2001.
〔17〕 M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem," In Proceedings of the AAAI National Conference on Artificial Intelligence, pp. 593-598, Alberta, Canada, Jul.-Aug. 2002.
〔18〕 I. Stewart, "Squaring the Square," Scientific American, Vol. 277, pp. 94-96, Jul. 1997.
〔19〕 U. Feige, "A Threshold of $\ln n$ for Approximating Set Cover," Journal of the ACM (JACM), Vol. 45, No. 4, pp. 634-652, Jul. 1998.
〔20〕 The Player Project, *http://playerstage.sourceforge.net/*

저 자 소 개

박 정 규

2002 : 홍익대학교 컴퓨터공학과 석사
2006-현재 : 홍익대학교 컴퓨터공학과 박사 과정
관심분야 : 지능형 로봇, 운영체제, 임베디드 시스템

전 흥 석

2001 : 홍익대학교 전자계산학과 박사
2007-2008 : 아이오와대학교 컴퓨터과학과 객원 교수
2002-현재 : 건국대학교 자연과학대학 컴퓨터 응용과학부 부교수
관심분야: 지능형 로봇, 임베디드 시스템

노 삼 혁

1986 : 서울대학교 컴퓨터공학과 학사
1993 : 매릴랜드대학교 컴퓨터과학과 박사
1993~1994 : 조지워싱턴대학교 객원조교수
1994-현재 : 홍익대학교 정보컴퓨터공학부 교수
관심분야 : 운영체제, 플래시메모리소프트웨어, 차세대저장장치