

축사 분뇨 처리 로봇의 효율적인 자율 작업을 위한 커버리지 최적 경로 계획

Optimized Coverage Path Planning for Efficient Autonomous Operation of a Barn Manure Handling Robot

지준구^{1,2}, 이명규², 김원균^{2,#}

Goo Jun Ji^{1,2}, Myeong Gyu Lee², and Won Gun Kim^{2,#}

¹ 한양대학교 대학원 전기공학과 (Electrical Engineering, Graduate School, Hanyang University)

² 한국생산기술연구원 (Korea Institute of Industrial Technology, KITECH)

Corresponding Author / E-mail: wgk@kitech.re.kr, TEL: +82-063-920-1296

ORCID: 0000-0003-2607-2128

KEYWORDS: Autonomous robot (자율주행 로봇), SLAM (동시적 위치추정 및 지도작성), Distance segmentation (거리 분할), Morphological segmentation (형태학적 분할), Voronoi segmentation (보로노이 분할), Coverage path planning (커버리지 경로 계획)

In the field of robotics and automation, path planning holds significant potential for optimizing field operations. These operations must cover the work area comprehensively and efficiently with minimal movement. To achieve these goals, coverage path planning (CPP) utilizing the Boustrophedon method is essential. However, in an experimental environment, CPP often results in missed work areas due to cumulative sensor errors and structural inconsistencies. This paper aimed to improve CPP by employing the Douglas-Peucker algorithm to simplify the work path and minimizing missed areas. Additionally, Edge Zone Path method was used to generate edge paths, enhancing safety of the trajectory. For experimental purposes, data were acquired from an actual barn. The work area was divided using three segmentation algorithms. Among these, the Voronoi Segmentation, which demonstrated superior performance, was used to extract the data. Experimental results indicated that the proposed optimized CPP improved path safety by maintaining a safe distance from obstacles during frequent turns. Additionally, the Coverage Ratio increased the coverage area of the autonomous robot by an average of 17% compared to the original CPP. These findings suggest that the proposed method can generate more efficient and safe work paths.

Manuscript received: May 31, 2024 / Revised: September 5, 2024 / Accepted: September 6, 2024

NOMENCLATURE

ε	=	Epsilon
DP	=	Douglas Peucker
CPP	=	Coverage Path Planning
W	=	Robot Width
L	=	Robot Length
τ	=	Turning Radius
NoS	=	Number of Segments

SA	=	Segment Area
SP	=	Segment Perimeter
NoE	=	Number of Edges
Nm	=	Normalized
INm	=	Inverse Normalized
NEZP	=	Number of Edge Zone Passes
EZP	=	Edge Zone Path
DIT	=	Design Infield Track
NoT	=	Number of Turns

1. Introduction

최근 축산업 인구 고령화 현상이 지속됨에 따라 자동화 기술의 필요성이 커지고 있다. 이에 따라 축산업 분야에서는 급이 로봇, 분뇨 처리 로봇에 대한 자율주행 연구가 활발히 진행되고 있다[1-3]. 특히, 자율주행 연구 중 경로 생성은 자율주행 작업의 효율성을 높이는 중요한 연구 주제로 자리 잡고 있다. 효율적인 작업 경로는 작업 시간 및 에너지 소모를 줄일 수 있다[4]. 따라서 작업 경로 효율성을 개선하고 최적의 작업 경로를 생성하는 것은 필수적이다.

현재 다양한 분야에서 작업 경로 생성 알고리즘인 CPP (Coverage Path Planning) 연구가 활발하게 진행되고 있다. 그러나 축산업 분야에서는 CPP 알고리즘과 같은 경로 생성 연구가 상대적으로 부족한 실정이다. 현재 축산업에서 개발된 방식은 장애물과의 접촉 이후 방향을 전환하여 주행하는 방식으로 경로 생성을 진행하고 있지 않다[5,6]. 이러한 방식은 주행의 일관성이 없으며, 잦은 중복 작업을 초래한다. 본 논문에서는 이러한 문제를 해결하기 위해 축산 분야에서 활용할 수 있는 CPP 기반 최적 작업 경로를 생성하고자 한다.

이를 위해, 본 논문에서는 한국농업기술진흥원에 설치되어 있는 축사시설에서 실험을 진행하였다. 알고리즘 개발 순서는 SLAM, 영역분할, 각 작업 영역에 대한 최적 CPP 알고리즘을 통한 작업 경로 생성 순으로 진행하였다. 먼저, 자율주행 로봇에 장착된 센서를 활용하여 축사 시설정보를 수집하였으며, 이를 SLAM을 통해 Point Cloud Map으로 변환하였다. 생성된 Point Cloud Map은 영역 분할을 위해 Grid Data로 변형시킨 후, OpenCV의 경계 검출 기능을 활용하여 축사시설의 경계 정보를 검출하였다[7]. 추출된 경계 정보 내부는 주행 가능 영역으로 지정하였으며, 이를 기반으로 영역 분할을 진행하였다. 영역분할은 축사시설의 외벽과 주행 가능 영역에 대한 정보를 이미지 형식으로 변환하여 활용하였다. 이를 활용하여, 세 가지 영역 분할 알고리즘(Distance Segmentation, Morphological Segmentation, Voronoi Segmentation)을 영역 분할 시간 Runtime, 분할된 영역의 개수 NoS (Number of Segments), 분할된 영역의 평균 넓이 SA (Segment Area), 분할된 영역의 평균 둘레 SP (Segment Perimeter), 그리고 분할된 영역의 가장자리의 개수 NoE (Number of Edges)를 취득 후, 이를 통해 결과를 비교하였으며, 결과적으로 Voronoi Segmentation 알고리즘이 본 논문에 활용하기에 가장 적합한 성능임을 확인하였다[8-11].

최적 CPP 알고리즘을 개발하기에 앞서 작업 영역 판별을 진행하였다. 작업 영역 판별은 분할된 각 영역의 이미지 정보를 바탕으로 로봇의 넓이 W (Robot Width)를 고려하여 작업이 필요 없는 영역을 소거하였다. 이를 통해 불필요한 작업 경로 생성을 방지할 수 있었다. 그러나, 이후 진행된 이미지를 Grid Data로 변형시키는 작업에서 측위 정보 및 센서의 노이즈로 인한 왜곡이 존재하여 기존 CPP 알고리즘을 활용하기에 어려움이 있었다. 이는 작업 경로 생성 시 비정형 패턴이 다수 발생하게

되는 문제점을 야기하기 때문이다. 작업 경로상에 존재하는 비정형 패턴은 작업의 효율성과 안전성을 저하시킨다. 따라서 본 논문에서는 DP (Douglas Peucker) 알고리즘을 작업 영역으로 판별된 Grid Data에 적용시켜, 허용 오차 ε (Epsilon) 이내의 불규칙한 성분을 제거하였다. DP CPP (Douglas Peucker Coverage Path Planning) 알고리즘으로 생성된 작업 경로는 기존 CPP 알고리즘을 활용한 작업 경로에 비해 단순하였으며, 이를 활용하여 일관성 있는 경로를 생성할 수 있었다. 또한, 작업 영역 내 경로의 효율성 지표인 Coverage Ratio가 평균적으로 증가한 것을 확인하였다.

기존 CPP 알고리즘은 선회 반경 τ (Turning Radius)에 대한 고려가 부족하다. 이는 작업 경로를 추종하는 과정에 제자리 선회와 같은 불필요한 동작을 유발할 수 있으며, 외벽과의 충돌 사고로 이어질 가능성이 존재한다. 본 논문에서는 이러한 위험요소를 줄이기 위해 τ 값을 고려하여 선회 구간 경로를 최적화하였다. 더불어, 주행 가능 영역 내부에 W 를 활용한 최소 회전 사각형을 구현하여 이를 기반으로 DP CPP 알고리즘을 구현하였다. 하지만 DP CPP 알고리즘은 기존 CPP 알고리즘을 활용하여 구현되었기 때문에, 작업 영역의 면적을 커버하는데 집중하는 특성이 있다[12]. 따라서 외벽 근처의 잦은 선회 구간에서 분뇨로 인한 미끄러짐 현상에 따른 충돌은 고려하지 않는다. 본 논문에서는 이러한 문제를 해결하기 위해 W , 가장자리 경로의 수 NEZP (Number of Edge Zone Passes)를 활용하여 가장자리 경로 EZP (Edge Zone Path)를 생성하였다. EZP는 Coverage Ratio를 증가시킬 뿐만 아니라, 잦은 선회 구간과 외벽 사이의 추가적인 간격을 확보하여 안전성도 향상시켰다. EZP에 DP 알고리즘을 적용한 경우(DP-EZP CPP)와 적용하지 않은 경우(NoN DP-EZP CPP)로 나누어 평가를 진행하였다. 두 가지 방법 모두 기존 CPP 및 DP CPP보다 Coverage Ratio가 증가하였으나, 둘 사이의 차이는 근소한 것을 확인하였다. 하지만 경로의 안전성 및 일관성 측면에서 선회 구간이 더 적게 생성되는 DP-EZP CPP가 우수한 것으로 확인하였으며, 최종적으로 최적 CPP는 DP-EZP CPP라는 결론을 얻었다. 이후, 생성된 각 영역을 연결하는 경로는 A* 알고리즘을 통해 연결하였다. 이를 통해 기존의 CPP의 문제점을 개선하여 효율적이고 안전성 있는 최적 작업경로를 생성하였다.

본 논문의 구성은 다음과 같다. 2장에서는 로봇의 제원 및 장착 센서를 설명한다. 3장에서는 실험 장소와 SLAM 활용 과정 및 결과를 다룬다. 4장에서는 주행 가능 영역 검출, 세 가지 영역분할 알고리즘에 대한 Runtime, NoS, SA, SP, NoE 적용 평가 및 결과에 대해 설명하였으며, DP 알고리즘에 대한 설명 및 적용 결과도 포함한다. 5장에서는 기존 CPP, DP CPP, Non DP-EZP CPP, DP-EZP CPP 알고리즘에 대해 설명하고 작업 경로 생성 결과를 보여준다. 이후, 최적 CPP 알고리즘인 DP-EZP CPP 알고리즘으로 구현된 작업 경로 사이의 연결을 위해 A* 알고리즘을 활용한 결과를 보여준다. 6장에서는 경로 생성 시간 Runtime, 생성된 경로의 길이 Length, 선회 구간의 개수 NoT

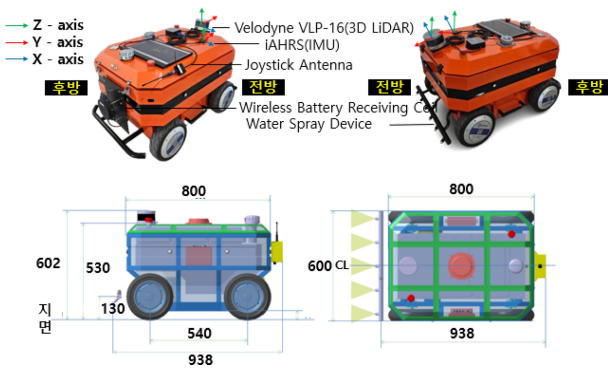


Fig. 1 Specifications and mounted sensors of the manure handling robot for barns

(Number of Turns), Coverage Ratio 적용 평가 및 최종적인 결과에 대해 설명한다. 7장에서는 최적 CPP 알고리즘 연구 및 향후 연구에 대해 제시한다.

2. Robot Description

Fig. 1은 실험에서 활용한 로봇에 대한 규격과 센서에 관한 자료를 보여준다. SLAM을 활용하기 위한 측위 정보를 취득하기 위해, 3차원 LiDAR인 Velodyne VLP-16 LiDAR 센서와 관성 측정 장치인 iAHRS IMU 센서를 장착하여 실험하였다. 로봇은 전방에 분뇨 및 소독 작업을 위한 분사 노즐이 장착되어 있어, 이를 사용하여 고압 물 분사 방식을 통해 작업을 진행하며, 스키드 제어 방식을 활용한다.

3. SLAM

3.1 HDL Graph SLAM

작업 경로 생성을 위한 사전 작업으로 ROS (Robot Operating System) 환경에서 Point Cloud Map을 제작하였다. 현재 위치추정 및 Mapping에 사용되고 있는 SLAM 알고리즘은 LOAM, A-LOAM, ICP, BLAM, HDL Graph SLAM 등 다양한 알고리즘이 사용되고 있다[13].

이 중, 본 논문에서는 ROS (Robot Operating System) 환경에서 사용 가능한 HDL Graph SLAM을 활용하였다[14]. HDL Graph SLAM은 센서 융합을 활용한 그래프 기반의 효율적인 접근 방식을 사용하며, FAST_GICP, FAST_VGICP와 같은 다양한 Scan Matching을 통해 Point Cloud Map 생성이 가능하다[15]. 이를 활용하기 위해 로봇에 장착되어 있는 센서를 활용하였으며, 결과적으로 HDL Graph SLAM을 활용하여 Fig. 2와 같이 한국농업기술진흥원에 설치되어 있는 축사시설의 Point Cloud Map을 구현하였다. 그러나 축사 시설은 주거 시설과 달리 평탄하지 않은 바닥면을 가지고 있다. 이로 인해 데이터 취득 시

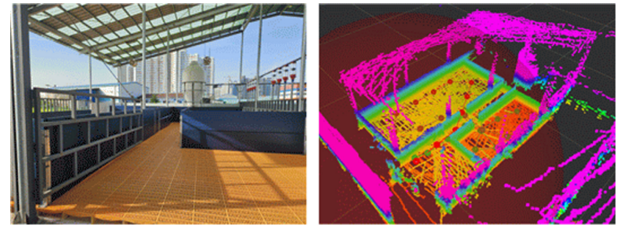


Fig. 2 Representation of a Point Cloud Map of a barn facility

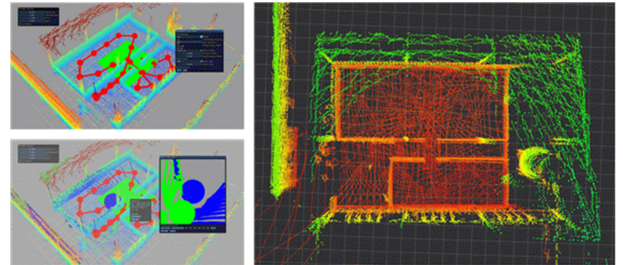


Fig. 3 Interactive SLAM based point cloud map

센서에 진동 현상이 발생하여 Point Cloud Map에 다수의 왜곡된 데이터가 생겼다. 이러한 문제를 해결하기 위해, 본 논문에서는 3.2장에서 소개한 Interactive SLAM을 활용하여 Point Cloud Map을 보정하였다.

3.2 Interactive SLAM

Interactive SLAM은 3D LIDAR 기반의 Frame Work로 Point Cloud Data를 활용하여 사용자가 수동으로 손상된 주행 거리, 잘못된 루프 감지, 왜곡된 Map Data와 같은 Mapping 오류를 직관적으로 수정할 수 있다. 또한 ICP, GICP, NDT, NDT OMP와 같은 Point Cloud Matching 알고리즘들을 통해 오류 수정을 할 수 있다[16]. 본 논문에서는 Mapping 오류를 수정하기 위해 오차가 가장 적은 Point Cloud 결과물을 활용하였으며, Interactive SLAM을 활용하여 일정 구간의 데이터를 임계값 설정을 통해 보정하였다. Fig. 3은 Interactive SLAM을 활용한 결과이다.

4. Area Division

4.1 3D Map to 2D Map Conversion

작업 영역을 검출하기 위해 3차원 정보인 Point Cloud Map 데이터를 사용하였다. x축은 -1.2에서 11.6 m, y축은 -5.2에서 6.8 m, z축은 0.0에서 0.5 m로 설정하여 유효한 영역을 추출하였으며, 그 결과는 Fig. 4에 나타나 있다. 포인트 검출에는 K-D Tree 알고리즘을 적용하여, 각 포인트를 기준으로 반경 0.1 m 이내의 근처 포인트들을 찾아 표면 법선 벡터를 계산하였다[17].

이후, 각 포인트의 법선 벡터를 활용하여 Grid Data를 생성하였다.

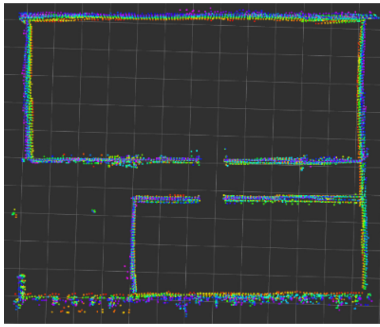


Fig. 4 Results of filtering the point cloud map by applying the K-D Tree algorithm

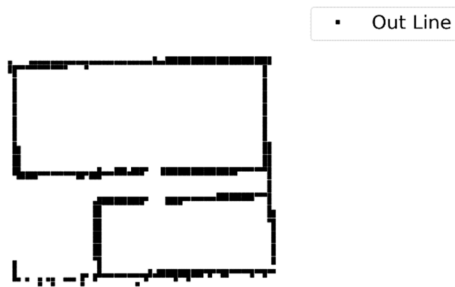


Fig. 5 Point cloud map of converted grid data & out line points

점유 Grid의 해상도는 0.2 m로 설정하고, 포인트의 좌표를 사용해 그리드 셀의 위치를 계산하였다. 계산 결과를 통해 특정 포인트의 x, y 좌표를 그리드 셀 위치로 변환하였다. 이후, 각 셀에 포함된 포인트 수를 추출하여 점유 상태 정보를 취득하였다. 셀에 포함된 점유 상태 포인트 수가 4 미만이면 해당 셀은 비어 있다고 간주하고, 포인트 수가 4 이상이면 점유된 셀로 표시하였다. Fig. 5는 점유된 셀을 Out Line 포인트로 나타낸 Grid Data 이미지이다.

4.2 Drivable Area Detection

Fig. 3의 이미지와 같이 본 논문에서 실험을 진행하고 있는 축사시설 내부에는 장애물이 존재하지 않는다. 따라서 내부 장애물에 대한 필터링 과정을 진행하지 않고, 축사시설에 대한 정보만 활용하여 주행 가능 영역을 판별하였다. 우선, 축사시설 내부 객체의 가장자리 좌표 4개를 추출하였다. 이 좌표들을 활용하여 OpenCV 경계 검출 기능을 통해 주행가능 영역의 경계 좌표를 검출하였다. 그 결과는 Fig. 6에서 확인할 수 있다. Fig. 6에서 Out Line은 외벽에 대한 좌표 정보를 나타내고, Inner Line 경계 좌표 내부 영역은 주행가능 영역을 의미한다. 결과적으로, Grid Data를 활용하여 주행 영역을 감지할 수 있는 알고리즘을 구현하였다.

이후, 4.3의 영역 분할 알고리즘에 활용하기 위해 Fig. 7과 같이 주행가능영역인 주행가능 영역인 Inner Line 포인트 내부는 흰색 영역, 외벽인 Out Line 포인트의 경우 검은색 영역, 그 외의 불필요한 영역은 회색 영역으로 설정하여 이미지화를 진행하였다.



Fig. 6 Drivable area determination based on contour line detection using OpenCV

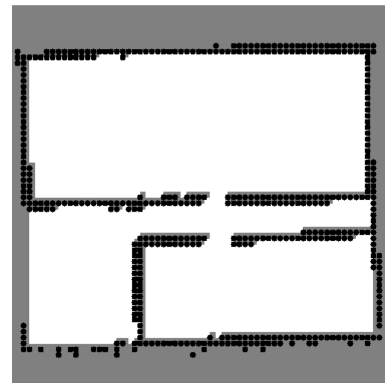


Fig. 7 Drivable areas in white, obstacles in black, and other areas in gray

4.3 Comparison of Area Partitioning Algorithms

현재 다양한 영역 분할 알고리즘들이 개발되어 있다. 그러나 가상환경에서의 영역 분할 실험은 많이 존재하지만, 축사 시설과 같은 실제 환경에서의 영역 분할 알고리즘 비교 연구는 부족한 실정이다[9-11]. 따라서 본 논문에서는 축사 시설의 데이터를 활용하여 자주 사용되는 세 가지 영역 분할 알고리즘에 대한 결과를 평가하고, 이를 통해 가장 적합한 알고리즘을 확인하여 최적 CPP에 활용하고자 한다. 세 가지 영역분할 알고리즘은 앞서 구현된 Fig. 7 이미지를 활용하여 진행하였으며, 각각의 알고리즘에 대한 설명 및 구현 결과는 다음과 같다.

A. Distance Segmentation

Distance Segmentation 알고리즘은 초기 이미지의 픽셀 간 거리 변환 데이터를 기반으로 한다. 이 알고리즘은 주행 가능한 영역인 흰색 픽셀과 가장 인접한 외벽인 검은색 픽셀 간의 거리를 측정하는 과정을 포함한다. 이 과정을 수행하기 위해 Fig. 7을 활용하였다. 변환된 거릿값 중 가장 큰 로컬 최댓값은 주행 가능한 영역의 중심에 위치하며, 좁은 영역이나 입구에서는 내부의 로컬 최댓값보다 작게 나타난다.

이 알고리즘은 거리 데이터를 활용하여 영역을 분할하므로, 임계값을 설정하여 영역 중심 정보를 추출해야 한다. 임계값을 적용하면 거리 데이터가 내림차순으로 정렬되고, 주행 가능 영역 중심의 크기가 증가한다. 임계값이 외벽 또는 입구에서의



Fig. 8 Distance segmentation is the result of area division

로컬 최댓값과 일치하면 주행 가능 영역 중심 수가 감소하기 시작하는데, 이는 인접한 주행 가능 영역 중심이 입구를 통해 병합되기 때문이다.

따라서, 주행 가능 영역의 중심 수가 최대가 되는 임계값을 선택하는 것이 필요하다. 최종적으로 결정된 각 영역의 중심에는 고유한 레이블을 부여하고, 나머지 공간은 레이블 없이 처리한다. 이러한 결과는 Fig. 8에 나타나 있다. Fig. 7의 이미지를 분석한 결과, 임계값 설정에 따라 영역이 분리되는 과정이 다름을 확인하였다. 다양한 임계값을 설정하여 Distance Segmentation 알고리즘을 적용한 결과, 4개의 세그먼트로 분할이 가능하였으나, SA 161.56 m², SP 59.22 m², NoE 8로 값이 높게 나타났다. 이러한 결과는 본 연구의 목적에 부합하지 않다고 판단하였다. 이는 각 작업 영역이 명확하게 구분되지 않음을 의미하며, 본 연구의 목적에 부합하지 않는 것으로 판단되었다. 본 연구에서는 더 세밀한 영역 분할이 필요하므로, 세분화가 충분하지 않은 Distance Segmentation은 적합하지 않은 것으로 결론지었다. 비록 이 알고리즘이 다른 알고리즘보다 구현 속도가 빠르다는 이점이 있지만, 본 연구에서는 정확한 분할이 더 중요한 요소이기 때문에 연구 목적에 맞지 않다고 판단하였다.

B. Morphological Segmentation

Morphological Segmentation 알고리즘은 단순성을 주요 특징으로 한다. 이 알고리즘을 활용하기 위해 Fig. 7의 이미지를 Grid Data로 변환한 픽셀값을 사용한다. 초기에는 외벽과 같은 장애물을 주행 불가능한 영역으로 레이블을 지정한다. 이 과정에서 흰색 픽셀은 주행 가능 영역, 검은색 픽셀은 외벽으로 나타낸다. Morphological Segmentation을 통해 각 픽셀에 반복적으로 레이블 작업을 수행한다.

이후, 주행 불가능 영역과 주행 가능 영역에 대한 구성 요소를 분석하여 주행 가능 영역이 분리되었는지 확인한다. 분리된 영역이 일정 크기, 즉 설정된 기준과 부합하면 모든 셀이 개별 영역으로 레이블 지정되고, 다른 영역과 분리되어 추출된다. 이 절차는 모든 영역의 셀이 제거될 때까지 반복된다. 이후, 레이블이 지정된 영역은 다른 영역과 레이블이 없는 공간으로 확장되어,



Fig. 9 Morphological segmentation is the result of area division

모든 주행 가능한 셀이 레이블로 지정된다.

결과적으로, 주행 가능한 셀과 주행 불가능한 셀을 기준으로 영역 분할이 이루어졌으며, 그 결과는 Fig. 9에 제시되어 있다. 실제 이미지 결과를 분석한 결과, 3개의 세그먼트로 분할하는 것이 가능했지만, SA 215.41 m², SP 67.99 m², NoE 6.25로 높은 값을 보였다. NoE 값은 Distance Segmentation보다 낮았지만, 다른 모든 지표에서는 전반적으로 기대에 미치지 못하였다. 이러한 결과는 4.4절에서 언급한 작업 판별 과정에서 각 작업 영역이 명확하게 구분되지 않는 문제를 초래한다. 따라서, 본 연구에서는 더욱 세밀한 영역 분할이 필요하다고 판단하여, Morphological Segmentation은 본 연구에 적용하기에 적합하지 않다고 판단하였다.

C. Voronoi Segmentation

Voronoi Segmentation 알고리즘은 Voronoi 다이어그램을 기반으로 분할을 진행한다. 먼저, Fig. 7을 활용한 Grid Data를 일정한 크기의 격자로 나누는 것으로 시작한다. 이 격자를 활용하여 각 셀에 대해 외벽 또는 주행 가능한 공간 정보를 토대로 Voronoi 다이어그램을 생성한다. 이후, 검출된 외벽 셀 간의 거리를 계산하여 Voronoi 경계를 설정하고, 이 경계를 활용하여 불필요한 세부 경계를 제거한다.

이를 통해 전체적인 Voronoi 다이어그램이 단순화된다. 다음으로, 외벽에서 검출된 가장 가까운 픽셀을 가진 점을 중요한 점으로 간주하고, 이 점들을 활용하여 주행 가능 영역을 분할하는 기준으로 설정한다. 이 점들을 활용하여 각각 연결하는 선들을 생성하고, 이 선을 통해 영역을 분할한다. 이 선들 중 사이각이 작은 각도를 갖는 것들은 모두 제거된다. 큰 각도를 갖는 선은 주로 주행 가능 영역의 입구 쪽에 표현된다.

하지만 이러한 분할 영역 중 일부는 너무 작아서 개별적으로 의미 있는 공간으로 간주하기 어려웠다. 이 문제를 해결하기 위해 휴리스틱을 적용하여 작은 영역을 큰 영역에 결합하였다. 휴리스틱 방법을 통해 영역 병합을 진행한 결과, Fig. 10에 제시된 바와 같이 최종적으로 6개의 세그먼트로 세분화된 이미지를 얻을 수 있었다. 이 결과는 세 가지 알고리즘 중 가장



Fig. 10 Voronoi Segmentation is the result of area division

Table 1 This is a table comparing the performance of three region division algorithms

	Distance	Morphological	Voronoi
Runtime[s]	2.07	5.91	15.04
NoS	4	3	6
SA [m ²]	161.56	215.41	107.70
SP [m]	59.22	67.99	36.78
NoE	8	6.25	4

세밀하고 명확하게 영역을 분할한 것으로, 본 연구에서 요구하는 바에 가장 적합한 형태임을 보여주었다. 구체적으로, SA 107.70 m², SP 36.78 m², NoE 4로, 다른 알고리즘에 우수한 성능을 보여주었다.

이러한 결과는 4.4절에서 설명한 작업 판별 과정에서 각 작업 영역이 명확히 구분되어야 하는 사항을 충족시키는 것으로, 본 연구에 활용하기에 가장 적합한 영역 분할 알고리즘임을 확인할 수 있었다.

4.4 Comparison and Analysis

다음 Table 1은 세 가지 알고리즘을 비교 분석한 결과이다. 비교 항목은 Runtime, NoS, SA, SP, NoE이다. 분석 결과, Distance Segmentation 방식이 가장 짧은 Runtime을 보였다. 그러나 NoS, SA, SP, NoE를 통한 정밀한 영역 분할 결과는 Voronoi Segmentation 방식이 가장 우수하였다.

Distance Segmentation 알고리즘은 경로 생성 시간에서 가장 효율적이었지만, 분할된 영역의 개수와 정밀도에서는 한계가 있었다. 반면, Morphological Segmentation 알고리즘은 단순한 구현과 빠른 처리 속도를 제공했으나, 분할된 영역의 정밀도와 정확성에서 부족했다. Voronoi Segmentation 알고리즘은 전체적으로 가장 정밀한 영역 분할을 수행했으며, 각 지표에서 균형 잡힌 성능을 나타냈다.

정확한 비교를 위해 각 요소에 가중치를 적용하여 각 알고리즘의 효율성에 대해 판단하였다. 이 중, 가장 중요하게 고려된 요소는 NoS와 NoE로 분할된 영역의 단순화 정도와 세분화 정도를 판단할 수 있기 때문이다. 따라서 NoS 가중치 w_{NoS} 와 NoE

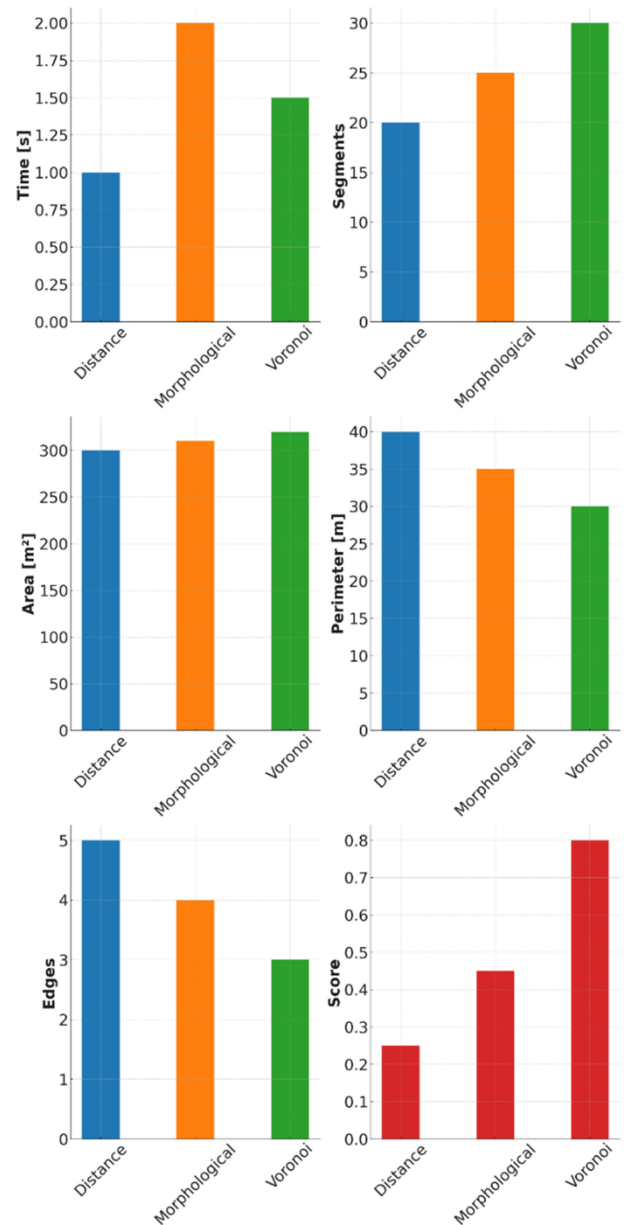


Fig. 11 Graph of segmentation for three methods evaluation results

가중치 w_{NoE} 는 0.3으로 설정하였다. 이외에 SA와 SP의 경우 면적의 넓이 및 둘레를 통해 영역의 세분화 정도를 추측할 수 있지만, 단순화 정도를 파악하지 못함으로 SA 가중치 w_{SA} 와 SP 가중치 w_{SP} 는 0.15로 설정하였다. Runtime 가중치 w_{Run} 는 0.1로 설정하였다. Runtime, SA, SP, NoE의 경우 수치가 낮을수록 영역분할 효율성이 증가함으로 식(2)을 통해 역 정규화 과정 INm (Inverse Normalized)을 진행하였다.

이에 반해 NoS의 경우 수치가 높을수록 영역 분할 효율성이 증가함으로 식(1)을 통해 정규화 과정 Nm (Normalized) 값을 도출하였다. 이후, 역정규화 수치를 정규화 수치를 정해진 가중치와 곱하고 최종적으로 모든 값을 합산한 수치인 Score로 영역 분할의 효율성으로 표현하였다.

Fig. 11은 앞서 설명한 수식을 통해 최종적인 결과를 도출한

것이며, Voronoi Segmentation의 최종적인 All Score 수치가 0.9로 가장 높은 것을 확인할 수 있다. 따라서 본 논문은 Voronoi Segmentation 과정을 통해 취득된 영역분할 데이터를 활용하였다.

$$Nm = \frac{Original\ Value - Min\ Value}{Max\ Value - Min\ Value} \quad (1)$$

$$INm = 1 - \frac{Original\ Value - Min}{Max\ Value - Min\ Value} \quad (2)$$

$$Runtime\ Score = (w_{Run} \times INm\ runtime) \quad (3)$$

$$Nos\ Score = (w_{NoS} \times Nm\ NoS) \quad (4)$$

$$SA\ Score = (w_{SA} \times INm\ SA) \quad (5)$$

$$SP\ Score = (w_{SP} \times INm\ SP) \quad (6)$$

$$NoE\ Score = (w_{NoE} \times INm\ NoE) \quad (7)$$

$$All\ Score = (3) + (4) + \dots + (7) \quad (8)$$

4.5 Workspace Filtering

본 절에서는 구현된 영역 분할 정보를 바탕으로 작업이 필요한 영역을 필터링하였다. 이를 위해 각 분할 영역의 면적을 측정하고, 이를 W와 비교하였다. 각 분할 영역의 가로 및 세로 길이를 추출하여 면적을 계산한 후, W의 2배와 비교하였다. 만약 분할된 영역의 면적이 W의 2배보다 작을 경우, 해당 영역을 제거하도록 알고리즘을 구현하였다. 이 알고리즘을 통해 불필요한 영역이 Fig. 12와 같이 소거되었다. 최종적으로, 작업이 필요한 영역의 경계 좌표를 추출하였다. 이를 통해, 불필요한 영역을 효과적으로 필터링함으로써, 후속 작업의 효율성을 높일 수 있었다.

4.6 Douglas-Peucker (DP) Algorithm

경계 좌표 검출 시 실시간 센서 데이터의 노이즈로 인해 다수의 왜곡된 경계 좌표 데이터가 검출되었다. 영역 분할 방식을 활용하는 다양한 알고리즘들은 이러한 왜곡요소를 충분히 고려하지 않으며, 이로 인해 경로 생성 과정에서 짧은 시간 내에 제어 변동성이 큰 경로 좌표가 생성되는 문제가 발생한다. 이는 주행의 일관성과 안전성에 부정적인 영향을 미친다.

본 논문에서는 이러한 문제점을 해결하고자 경계 좌표의 기존 곡선 특성을 유지하면서 경계 좌표를 단순화할 수 있는 DP 알고리즘을 활용하였다[16]. Fig. 13과 같이 DP 알고리즘은 경계 좌표의 각 점과 기준선 사이의 거리를 측정하여 최대 수직 거리를 갖는 점을 찾아내고, 이 점과 기준선 간의 거리가 ϵ 보다 작으면 해당 점을 제거한다. 이 과정을 끝점 사이에 점이 없을 때까지 반복한다.

아래의 수식은 DP 알고리즘과 관련된 수식을 설명한 것이다. 식 (9)에서 주어진 것은 시작 곡선 p 이며 이는 n 개의 점으로 이루어진

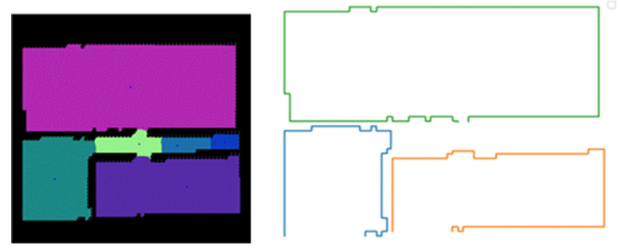


Fig. 12 As a result of dividing the work area, three areas were detected

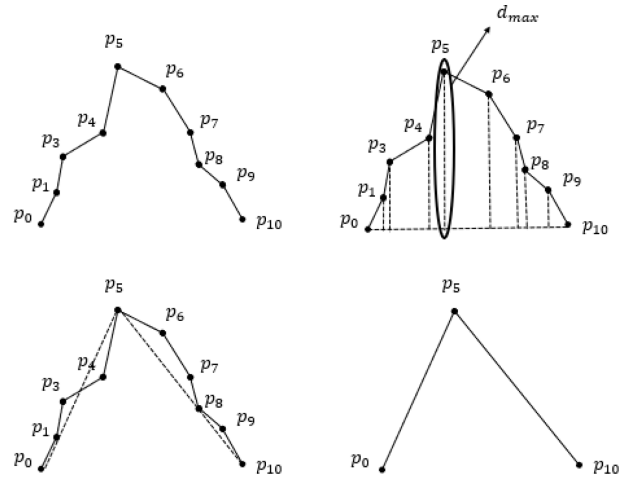


Fig. 13 A simple image of the douglas-peucker algorithm shows that unnecessary coordinates in p have disappeared.

정렬된 집합으로 D의 첫 번째 근사치 선분을 p 로 표현하였다. 식 (10)과 같이 모든 $n-2$ 개의 내부 점을 확인하고, p_0 와 p_n 를 잇는 점선과 P_i 사이의 가장 먼 점을 d_{max} 수치로 산출하였다. 만약 d_{max} 가 허용 오차인 ϵ 이내에 있다면, 허용 오차 내의 점들을 제거하는 단순화 작업을 진행하였다.

결과적으로 DP 알고리즘을 통해 작업 영역의 경계 좌표 중 일부를 선택하고, 미리 정의된 ϵ 내에 있는 점들을 소거하여 단순화된 경계 좌표를 추출하였다.

$$D = (p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}) \quad (9)$$

$$d_{max} = \max_{i=2 \dots n-1} d(p_i, \overline{p_1 p_n}) \quad (10)$$

$$d_{max} \leq \epsilon(1.0) \quad (11)$$

$$D_1 = (p_0, p_5, p_{10}) \quad (12)$$

Fig. 14는 검출된 경계 좌표에 DP 알고리즘을 적용한 모습을 보여준다. 이를 통해 경로 생성을 위한 경계 좌표의 왜곡된 부분을 선형 구조로 단순화하였으며, 경로 생성 시 발생할 수 있는 비정형 패턴을 감소시켰다. DP 알고리즘을 적용한 결과와 적용하지 않은 작업 경로 생성 결과는 Fig. 20에서 확인할 수 있다.

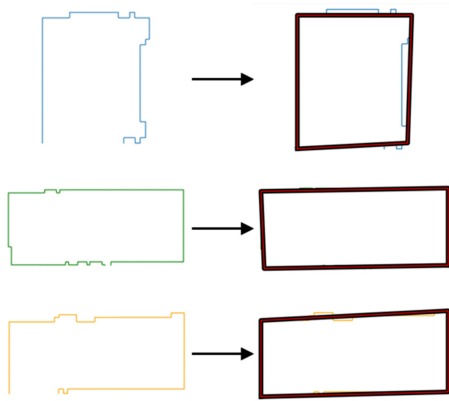


Fig. 14 Results of applying douglas-peucker algorithm to three areas

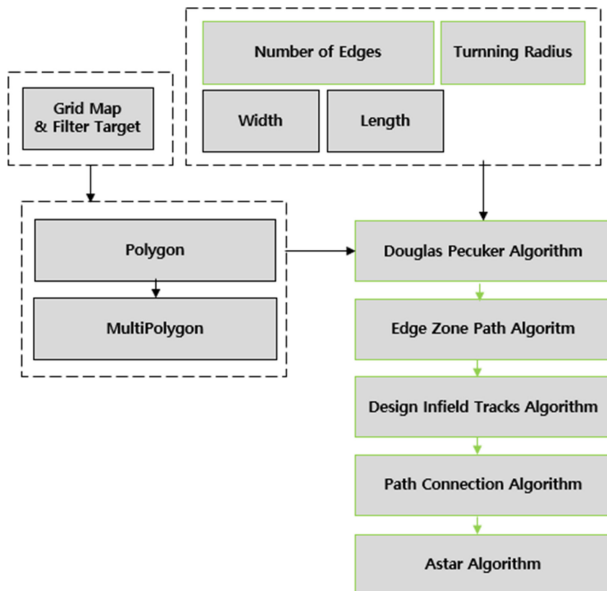


Fig. 15 Architecture for path planning, explaining the overall path creation sequence

5. Coverage Path Planning

5.1 Coverage Path Planning Architecture

Fig. 15는 작업 경로 생성 알고리즘의 개략도를 나타낸다. 4.4절에서 처리된 결과물을 활용하여 경로 생성을 진행하였다. 구현된 결과물을 데이터 처리에 용이한 폴리곤(Polygon) 형태로 변환하였으며, 장애물이 존재할 수 있는 상황에 대비하여 멀티폴리곤(MultiPolygon) 형태로도 변환하였다.

경로 생성을 위한 주요 파라미터는 다음과 같다. NEZP 값은 1로 설정하였으며, τ 의 설정 값은 0.1 m로 설정하였다. W의 설정 값은 0.6 m이며, 로봇의 길이 L (Robot Length)의 설정 값은 0.8 m로 설정하여 경로 생성을 진행하였다.

경로 생성 과정 중 EZP는 파라미터 값을 모두 활용하여 효율적인 가장자리 경로를 생성하였다. 이후, DIT (Design Infield

Tracks)는 가장자리 경로의 내부 작업 경로를 생성하는 과정으로 W를 활용하였다. 생성된 경로는 Path Connection 과정을 통해 연결하였으며 τ 를 활용하여 완만하 선회 경로를 생성하였다. 이를 통해 작업을 위한 최적 경로가 생성되었다. 이후, 각 영역의 작업 경로는 A*알고리즘을 통해 연결하였다.

결과적으로 기존 CPP 알고리즘 방식과 달리 DP, EZP, DIT를 통해 더 안전성과 효율성 있는 최적의 CPP 알고리즘을 개발하였다. 구체적인 알고리즘의 상세 내용은 5.2-5.4장에서 설명하였다.

5.2 Edge Zone Path (EZP) Algorithm

본 논문에서 제안하는 EZP는 생성하여 주행의 안전성을 확보하고 작업의 효율성을 높이는 경로 생성 과정이다. 기존 CPP 알고리즘은 주행의 안전성을 고려하지 않고, Coverage Ratio에만 중점을 두고 있다.

그러나 축사 시설 환경에서는 분노로 인한 미끄러짐 현상이나 계절에 따른 지면 변화 등 다양한 변수가 발생한다. 이러한 상황으로 인해 로봇은 외벽과의 충돌 현상이 발생한다. 본 논문에서는 이러한 문제를 해결하기 위해 가장자리 경로 EZP를 도입하였으며 구현은 다음과 같다.

4.4절에서 처리된 결과물인 x, y 좌표를 식(13)와 같이 Polygon 형식으로 변환하여 데이터를 활용하였으며, 이를 통해 각 변의 정보를 취득하였다. 이후, 각 변에 대한 정보를 L_i 로 지정하고 로봇의 폭 W 만큼 평행한 오프셋 경로를 식(14)과 같이 구현하였다. 수식에 사용되는 벡터 $\hat{n}_i$ 는 변 L_i 에 대한 각 변의 수직인 단위 벡터이다. 구현된 각 변의 평행한 오프셋 경로는 식(15)의 수식과 같이 결합하여 가장자리 경로를 형성하였다.

$$Polygon = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\} \quad (13)$$

$$L_i^{offset} = L_i + W \times \hat{n}_i \quad (14)$$

$$Edge Zon Path = \bigcup_{i=1}^n L_i^{offset} \quad (15)$$

기존의 CPP 알고리즘과 달리, 본 논문에서 제안하는 가장자리 경로 생성은 주행의 안전성을 확보하는 데 중요한 역할을 한다. Fig. 16은 이러한 가장자리 경로 생성의 결과를 보여준다. 최종 결과인 Fig. 20을 확인하면, 잦은 선회 구간이 EZP로 인해 기존보다 외벽과의 간격을 확보하여 내부에 생성되어 안전성을 확보하는 모습을 확인할 수 있다.

또한 기존 CPP 알고리즘을 통한 작업 경로 생성 결과보다 작업 영역 내 경로의 효율성 지표인 Coverage Ratio 수치가 증가한 모습을 보여주었으며, 이를 통해 작업의 효율성이 증가한 것을 확인할 수 있다. 이에 대한 결과는 6장에서 확인할 수 있다.

5.3 Design Infield Tracks (DIT) Algorithm

가장자리 경로 생성 이후, 내부 경로 생성 DIT는 가장자리 경로 내부에 경로를 생성하는 알고리즘이다. 생성 방식은 가장

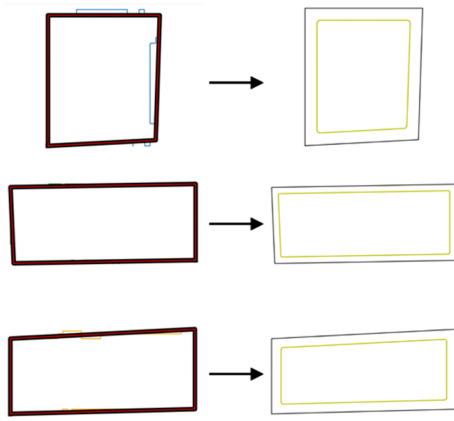


Fig. 16 Edge zone path is an edge path that ensures safety from frequent turning sections



Fig. 17 Detecting the longest axis by repeatedly rotating as much as θ using minrect

자리 경로 내부를 둘러싸는 최소 회전 사각형 MinRect를 구현하는 것이다. MinRect는 5.2의 방식으로 구현되었다. 5.3의 식(14)에 5.2에서 취득한 EZP 정보를 대입하여, 사각형을 구현할 수 있는 좌표를 탐색하고 식(16)와 같이 추출하였다. 이를 활용한 결과는 Fig. 18에서 가장자리 경로의 내부 사각형으로 확인할 수 있다. 이렇게 얻어진 MinRect의 각 변은 가장자리 경로 내의 내부 경로 생성을 위한 주요 축으로 활용되었다. 이후, 식(19)을 활용하여 구현된 MinRect를 θ 에 따라 회전시키며 가장자리 경로 내에 생성할 수 있는 가장 긴 변의 도형을 구현하였다. 구현 결과는 Fig. 18에서 확인할 수 있다.

$$\text{MinRect} = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)\} \quad (16)$$

$$d_1 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (17)$$

$$d_2 = \sqrt{(x_3 - x_4)^2 + (y_3 - y_4)^2} \quad (18)$$

$$\theta = \begin{cases} \tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right) & \text{if } d_1 > d_2 \\ \tan^{-1}\left(\frac{y_4 - y_3}{x_4 - x_3}\right) & \text{if } d_1 > d_2 \end{cases} \quad (19)$$

구현 결과, Fig. 17과 같이 d_1 이 가장 긴 축인 경우, 식(20)를 통해 x축 성분 x_n 과 y축 성분 y_{wd} 을 활용하여 P_x 경로를 생성하였다. 여기서 y축 성분 y_{wd} 는 로봇의 폭 W 를 활용하여 계산된다. 반면, d_2 가 가장 긴 축이라면 식(21)를 통해 y축 성분 y_n 과 x축 성분 x_{wd} 을 활용하여 P_y 경로를 생성하였다. 이때

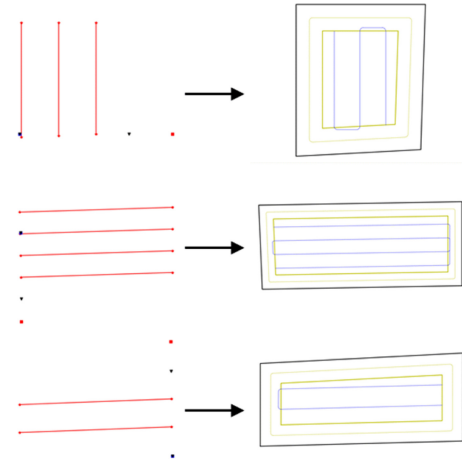


Fig. 18 Create a path within the edge zone path as a result of the design infield path

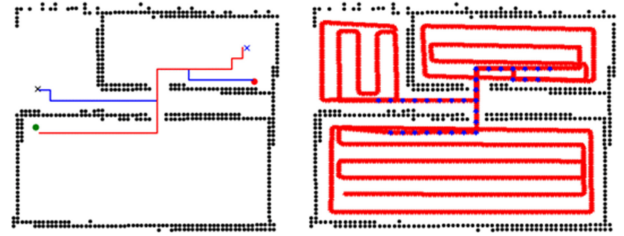


Fig. 19 A* is used to connect the coverage paths in each area to create the final working coverage path

x축 성분 x_{wd} 는 로봇의 폭 W 를 기준으로 계산된다.

이후, 식(22)를 통해 두 점의 중간 점인 P_z 를 추출하였다. 다음으로, 식(23)에 세 점 P_x , P_y , P_z 를 활용하였으며, τ 를 설정하여 완만한 선회 반경으로 조정하였다. 결과적으로 선회 반경을 고려한 가장자리 경로의 내부 작업 경로를 생성하였다. 최종적인 적용 결과는 Fig. 18에서 확인할 수 있다.

기존의 CPP 알고리즘에서는 비일관성 된 작업 경로가 생성되는 경향이 있었지만, 이 과정을 통해 일관성 있는 작업 경로를 생성하여 작업의 일관성 및 효율성을 향상시킬 수 있었다. 비교 결과는 Fig. 20에서 확인할 수 있다.

$$P_x = \{(x_n, y_{wd})\} \quad (20)$$

$$P_y = \{(x_{wd}, y_n)\} \quad (21)$$

$$P_z = \left(\frac{x_n + x_{wd}}{2}, \frac{y_{wd} + y_n}{2} \right) \quad (22)$$

$$\text{Inner } L_{\text{connect}} = \text{LineString}([P_x, P_y, P_z], \tau) \quad (23)$$

5.4 Path Connection Algorithm

5.3의 결과를 통해 생성된 가장자리 경로와 가장자리 경로의 내부 작업 경로를 연결해 주기 위해 Path Connection 작업을 진행하였다. 식은 5.3의 식과 동일하며, 가장자리 경로의 끝점 P_c 와

내부 경로의 시작점 P_s 을 활용하였다. 이 두 점의 중간 점인 P_m 을 추출하였다. 식(26)에 세 점 P_e , P_s , P_m 을 활용하여 가장자리 경로와 가장자리 내부의 작업 경로를 연결해주는 연결 경로를 생성하였다. 이후, 생성된 경로는 τ 를 설정하여 완만한 선회 반경으로 조정하였다. 최종적인 결과는 Fig. 20 (d) DP-EZP CPP 알고리즘의 결과를 통해 확인할 수 있다.

결과적으로 최종적인 작업 영역의 각 경로는 가장자리 경로에서 가장자리 내부의 작업 경로에 원활하게 진입할 수 있는 경로를 생성하였다. 이를 통해 분할된 영역에 대한 작업을 위한 최적화된 CPP 알고리즘을 생성하였다.

$$P_e = (x_e, y_e) \quad (24)$$

$$P_m = \left(\frac{x_e + x_s}{2}, \frac{y_e + y_s}{2} \right) \quad (25)$$

$$L_{connect} = LoneString([P_e, P_m, P_s] \tau) \quad (26)$$

5.5 Astar Algorithm

각 영역에 생성된 커버리지 작업 경로는 영역 사이를 연결해주는 주행 경로가 생성되어 있지 않다. 따라서 출발지점부터 목표 지점까지 이동할 수 있는 최적 경로를 생성하기 위해 A* 알고리즘을 활용하여 경로들을 연결하였다. A* 알고리즘에 사용된 값은 로봇의 폭(W)과 길이(L)를 반영하여 로봇에 맞춘 최적 경로를 구현하였다. 그러나 기본적인 A* 알고리즘을 사용하여 커버리지 작업 경로와 연결할 때 부적합한 경로가 생성되는 문제가 발생하였다.

이 문제를 해결하기 위해 A* 알고리즘에서 생성된 경로의 양쪽 끝부분 좌표를 일정 부분 소거하였다. 이후 남아 있는 좌표를 보간법을 활용하여 로봇이 주행하기에 적합한 형태로 경로를 생성하였다. 이를 통해 각 영역의 작업 경로들을 효과적으로 연결하였다.

Fig. 19는 A* 알고리즘을 통해 경로를 보여주며, 이를 통해 영역마다 문제없이 주행할 수 있는 효율적인 경로가 생성되었음을 확인할 수 있다.

6. Comparison of Generated Coverage Path

본 논문에서는 기존 CPP 알고리즘, DP 알고리즘을 적용한 DP CPP, DP 알고리즘을 적용하지 않고 EZP만 적용한 Non-DP EZP CPP, DP 알고리즘과 EZP를 모두 활용한 DP-EZP CPP를 비교하였다. 비교 항목은 각 영역에 대한 경로 생성 시간 Runtime, 생성된 경로의 길이 Length, Coverage Ratio이다. 각 알고리즘의 비교 항목에 가중치를 곱하여 각각의 Score를 산출하였다. 산출된 Score는 모두 합산하여 All Score 값으로 변환하였으며, 이를 통해 최종적인 최적 CPP 알고리즘을 비교하였다.

Fig. 20에서 볼 수 있듯이, 기존 CPP 알고리즘은 정형화된

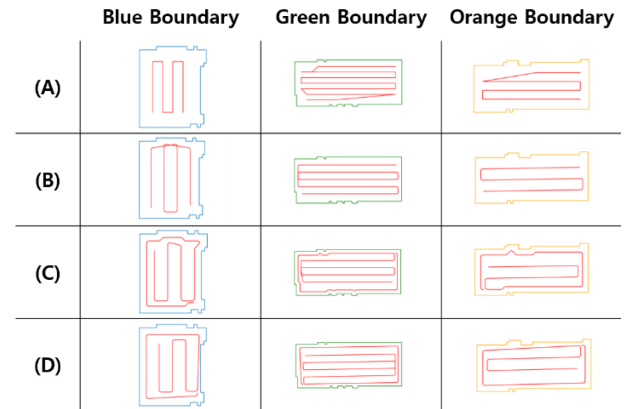


Fig. 20 Coverage path result: (A) Original CPP (B) DP CPP (C) Non DP-EZP CPP, (D) DP-EZP CPP

패턴을 제공하지 않았으며, 누락된 작업 영역이 존재하는 것을 확인하였다. DP CPP 알고리즘의 경우, 잦은 선회 구간이 외벽 근처에 생성되어 분노와 같은 환경적 요소 때문에 외벽과의 충돌 현상이 발생할 가능성이 있었다. 또한, Coverage Ratio가 충분하지 않은 것으로 확인하였다.

Non DP-EZP CPP 알고리즘의 경우, Coverage Ratio가 DP-EZP CPP 알고리즘과 유사하였지만, 선회 구간의 수가 NoT (Number of Turns) 증가했을 뿐만 아니라 급격한 선회 구간도 발생하였다. 따라서 이는 작업 경로로 활용하기에 어려움이 있을 것으로 판단된다. 반면, DP-EZP CPP 알고리즘은 정형화된 패턴의 경로를 제공하며, 선회 구간에서 외벽과의 충돌을 줄일 수 있는 가장자리 경로를 통해 안전성을 확보하였다. 또한, 잦은 선회 구간이 외벽과의 거리를 유지하면서 배치되어 로봇의 안전한 주행을 보장하였다.

이러한 결과는 DP-EZP CPP 알고리즘이 주행 경로의 일관성과 효율성을 향상시키는 데 중요한 역할을 한다는 것을 보여준다.

Fig. 21에서 Figs. 23은 20의 비교 결과를 나타낸 이미지이다. 4.3절에서 사용된 평가 방식과 유사한 방식으로 경로 생성 결과를 비교하였다. Blue, Green, Orange Boundary의 경로 생성 결과를 경로 생성 시간 Runtime, 작업 경로 길이 Path Length, NoT, Coverage Ratio 총 4개의 평가지표를 분류하여 비교하였다.

최종적인 평가는 All Score를 통해 성능을 평가하였다. 본 논문에서는 Coverage Ratio의 수치가 증가할수록 성능이 향상되므로 식(1)을 통해 정규화 작업을 진행하였다. 반면, Runtime, Path Length, NoT의 경우, 수치가 작을수록 성능이 향상되므로 식(2)을 통해 역으로 정규화하여 각 Score에 적용하였다.

Runtime의 가중치 w_r 와 Path Length의 가중치 w_l 는 작업 경로의 생성시간 및 길이에 대한 효율성을 고려하여 0.1로 설정하였다. NoT의 가중치 w_n 는 작업의 안전성을 고려하여 0.15로 설정하였다. Coverage Ratio의 가중치 w_c 는 할당된 작업 영역을 빠짐없이 커버할 수 있는 비율이 높을수록 좋은 성능을 나타내므로 0.65로 설정하였다. 본 논문에서는 일관성, 효율성, 그리고 안전성에 중점을 두었으며, 그 중 가장 중요한 요소로 Coverage

Table 2 The path creation evaluation results of blue boundary

Blue	Original CPP	DP CPP	Non DP-EZP CPP	DP-EZP CPP
Runtime [s]	0.07	0.37	2.51	2.47
Length [m]	15.24	16.67	17.81	17.24
NoT	6	6	14	10
Coverage [%]	79.365	67.3	96.913	98.748

Ratio를 설정하여 가장 높은 가중치를 설정하였다.

식(33)을 통해 취득된 정보와 설정 값을 기반으로 최종적으로 모든 값을 합산한 수치 Score를 산출하여 경로 생성 결과를 비교하였다.

$$Nm = \frac{\text{Original Value} - \text{Min Value}}{\text{Max Value} - \text{Min Value}} \quad (27)$$

$$INm = 1 - \frac{\text{Original Value} - \text{Min}}{\text{Max Value} - \text{Min Value}} \quad (28)$$

$$\text{Runtime Score} = (w_t \times Inm \text{ Time}) \quad (29)$$

$$\text{Length Score} = (w_L \times INm \text{ Length}) \quad (30)$$

$$\text{NoT Score} = (w_N \times INm \text{ Not}) \quad (31)$$

$$\text{Coverage Score} = (w_c \times Nm \text{ Coverage}) \quad (32)$$

$$\text{All Score} = (29) + (30) + (31) + (32) \quad (33)$$

Table 2는 Blue Boundary에 대한 비교 결과를 나타낸 표이다. Original CPP의 Runtime, Length, NoT의 성능이 가장 우수하였으나, Coverage Ratio가 Non DP-EZP CPP, DP-EZP CPP에 비해 현저히 낮은 것을 확인하였다.

Fig. 21는 Blue Boundary에 대한 비교 정보를 표현한 Table 2를 활용하여 그래프로 표현한 것이다. Score 결과가 Original CPP는 0.599, DP CPP는 0.282, Non DP-EZP CPP는 0.612, DP-EZP CPP는 0.748로 나타났다. Original CPP는 Runtime, Path Length, NoT에서 가장 우수한 성능을 보였으나, Coverage Ratio가 충분하지 않았으며, 결과적으로 누락된 작업 영역이 발생하였다. 따라서 본 논문에서는 Coverage Ratio가 가장 높은 Non DP-EZP CPP와 DP-EZP CPP 알고리즘의 결과값을 중점적으로 비교하였으며, 이 중 NoT가 작은 DP-EZP CPP 알고리즘이 가장 최적화된 작업 경로 생성 결과를 임을 확인하였다.

Table 3는 Green Boundary에 대한 비교 결과를 나타낸 표이다. Original CPP의 Runtime, Length의 성능이 가장 우수하였으며 NoT로 인한 안전성은 DP CPP가 가장 우수하였다. 하지만 두 경우 모두 Coverage Ratio가 Non DP-EZP CPP, DP-EZP CPP에 비해 낮은 것을 확인하였다.

Fig. 22는 Green Boundary에 대한 비교 정보를 표현한 Table 3을 활용하여 그래프로 표현한 것이다. Score 결과가 Original CPP는 0.275, DP CPP는 0.619, Non DP-EZP CPP는 0.627, DP-EZP CPP는 0.764로 나타났다. Original CPP는 Runtime, Path

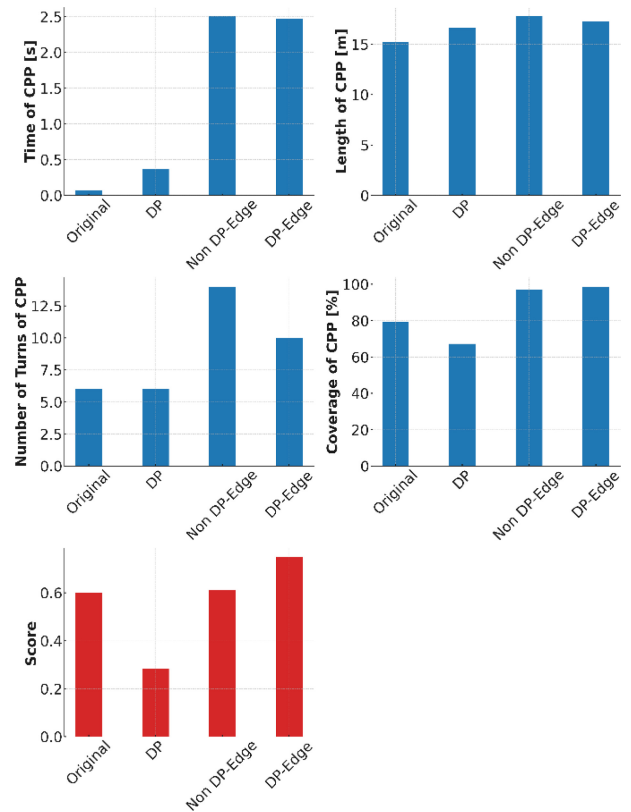


Fig. 21 Graph of path creation for blue boundary evaluation results

Table 3 This table expressing the path creation evaluation results of green boundary coordinates

Green	Original CPP	DP CPP	Non DP-EZP CPP	DP-EZP CPP
Runtime [s]	0.09	0.79	3.51	3.19
Length [m]	68.248	71.75	73.963	73.327
NoT	12	8	16	11
Coverage [%]	84.269	92.56	99.082	99.609

Length에서 가장 우수한 성능을 보였으나, Coverage Ratio가 충분하지 않았다. 앞선 결과와 마찬가지로 Non DP-EZP CPP와 DP-EZP CPP의 Coverage Ratio 값이 가장 높아 이 둘을 비교하였다. 하지만 Non DP-EZP CPP의 NoT 값이 높아 결과적으로 DP-EZP CPP가 가장 최적화된 작업 경로임을 확인하였다.

Table 4는 Orange Boundary에 대한 비교 결과를 나타낸 표이다. 앞선 두 Boundary 결과와 동일하게 Original CPP의 Runtime, Path Length가 가장 우수한 성능을 보였으나, Coverage Ratio의 수치가 낮아 작업 경로 활용하기 어려운 것으로 확인하였다.

Fig. 23은 Orange Boundary에 대한 비교 정보를 표현한 Table 4를 활용하여 그래프로 표현한 것이다. Score 결과가 Original CPP는 0.32, DP CPP는 0.665, Non DP-EZP CPP는 0.606, DP-EZP CPP는 0.744로 나타났다. Original CPP는 Runtime, Path

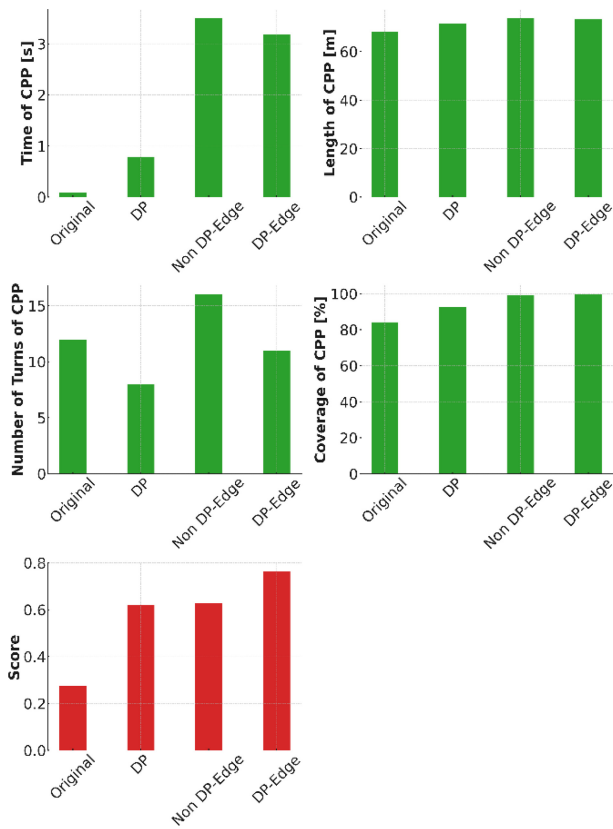


Fig. 22 Graph of path creation for green boundary evaluation results

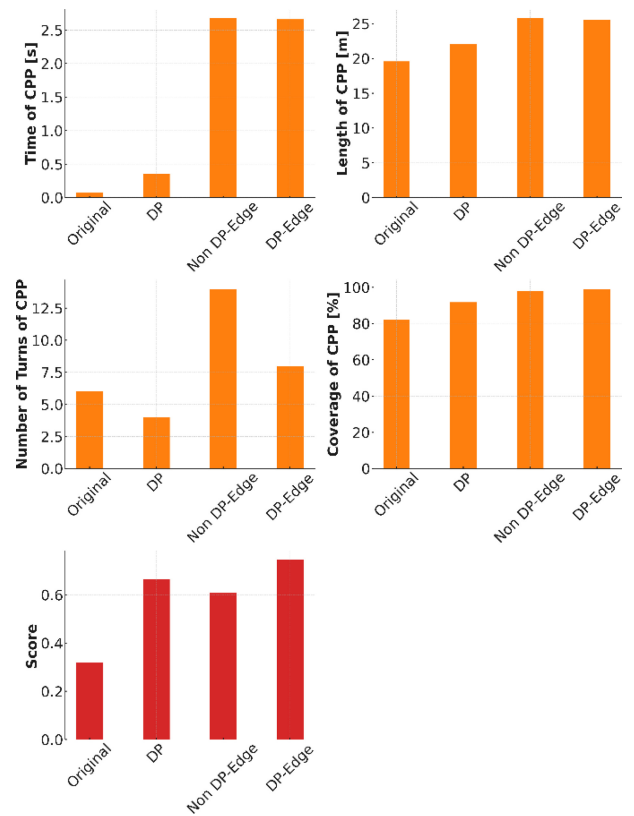


Fig. 23 Graph of path creation for orange boundary evaluation results

Table 4 The path creation evaluation results of orange boundary

Orange	Original CPP	DP CPP	Non DP-EZP CPP	DP-EZP CPP
Runtime [s]	0.07	0.36	2.68	2.66
Length [m]	19.57	22.07	25.824	25.584
NoT	6	4	14	8
Coverage [%]	82.25	91.78	98.028	99.158

Length에서 가장 우수한 성능을 보였으나, Coverage Ratio가 충분하지 않았다. 앞선 결과와 마찬가지로 Non DP-EZP CPP와 DP-EZP CPP의 Coverage Ratio 값이 가장 높아 이 둘을 비교하였다. 하지만 Non DP-EZP CPP의 NoT 값이 높아 결과적으로 DP-EZP CPP가 가장 최적화된 작업 경로임을 확인하였다.

7. Conclusion

본 논문에서는 세 가지 영역분할 알고리즘을 비교하여 실제 측사 환경에 활용할 수 있는 적합한 알고리즘을 탐색하고, 성능을 비교하여 최적의 영역분할 알고리즘을 판별하였다. 또한, 기존 CPP 알고리즘을 변형하여 측사 환경에 효율적이면서 안전성을 확보한 최적의 CPP 알고리즘을 개발하였다.

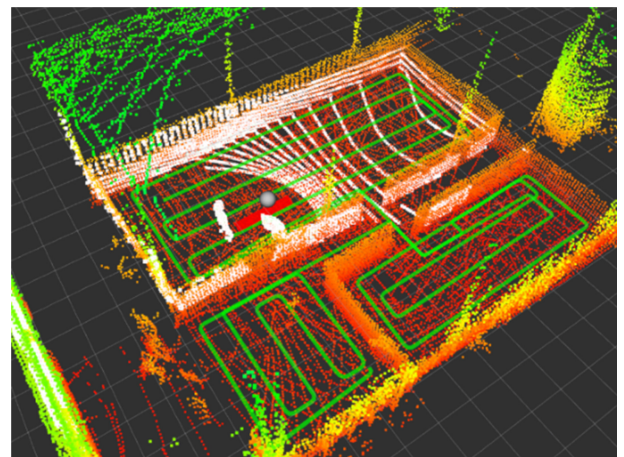


Fig. 24 Experimental method using the final coverage path creation results

세 가지의 영역 분할 알고리즘을 활용하여 성능을 비교한 결과, 실제 측사 환경에서 Voronoi Segmentation 알고리즘이 가장 우수한 성능을 보였다. 따라서 본 논문에서는 Voronoi Segmentation을 통해 영역 분할을 진행하였다.

기존의 영역 분할 데이터는 비정형적인 형태로 나타나므로, DP 알고리즘을 적용하여 데이터를 단순화하였다. 이를 통해 변동성이 큰 경로 좌표를 제거함으로써 주행의 일관성과 안전성을 확보하였다.

기존 커버리지 경로 계획은 환경적 요소와 잦은 방향 전환으로 인한 장애물 충돌을 고려하지 않는다. 본 논문에서는 이러한 문제를 해결하기 위해 측사의 외벽과의 충돌을 막고자 가장자리 경로를 생성하였다. 이를 통해 선회 구간과 측사 외벽 간의 거리를 확보하여 충돌을 최소화하였다. 또한 이를 통해 Coverage Ratio를 증가시켰다.

결과적으로 실제 측사 환경에서 활용 가능한 경로 계획을 개발하였으며, 생성된 커버리지 작업 경로는 효율성과 안전성을 확보하였다.

본 연구에서 생성된 커버리지 작업 경로는 경로 추종 알고리즘을 구현하여 Fig. 24와 같은 방식으로 성능을 평가할 예정이며, 이에 따른 로봇의 분포 처리 작업 효율을 평가할 것이다.

ACKNOWLEDGEMENT

본 논문은 농림축산식품부의 ‘차세대융합원천기술연구사업’의 지원을 받아 연구되었다(No. 421044-04).

REFERENCES

1. Tian, F., Wang, X., Yu, S., Wang, R., Song, Z., Yan, Y., Li, F., Wang, Z., Yu, Z., (2022), Research on navigation path extraction and obstacle avoidance strategy for pusher robot in dairy farm, *Agriculture*, 12(7), 1008.
2. Stasewitsch, I., Schattenberg, J., Frerichs, L., (2019), Cleaning robot for free stall dairy barns: sequential control for cleaning and littering of cubicles, *Robot 2019: Fourth Iberian Robotics Conference*, 115-126.
3. Shamshiri, R. R., Weltzien, C., Hameed, I. A., Yule, I. J., Grift, T. E., Balasundram, S. K., Pitonakova, L., Ahmad, D., Chowdhary, G., (2018), Research and development in agricultural robotics: A perspective of digital farming, *International Journal of Agricultural and Biological Engineering*, 11(4), 1-14.
4. Jin, J., Tang, L., (2010), Optimal coverage path planning for arable farming on 2D surfaces, *Transactions of the ASABE*, 53(1), 283-295.
5. Lely, Lely discovery series: robotic manure handling solutions, <https://www.lely.com>
6. JOZ, JOZ-Tech JT200 EVO Manure Robot, www.joz.nl/en
7. Sakshi, K. V., (2022), Segmentation and contour detection for handwritten mathematical expressions using OpenCV, *Proceedings of the International Conference on Decision Aid Sciences and Applications (DASA)*, 305-310.
8. Chen, Z., Qi, Z., Meng, F., Cui, L., Shi, Y., (2015), Image Segmentation via Improving Clustering Algorithms with Density and Distance, *Procedia Computer Science*, 55, 1015-1022.
9. Frias, E., Balado, J., Díaz-Vilariño, L., Lorenzo, H., (2020), Point cloud room segmentation based on indoor spaces and 3D mathematical morphology, *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 44, 49-55.
10. Kise, K., Sato, A., Iwata, M., (1998), Segmentation of page images using the area voronoi diagram, *Computer Vision and Image Understanding*, 70(3), 370-382.
11. Jung, J., Stachniss, C., Kim, C., (2017), Automatic room segmentation of 3d laser data using morphological processing, *ISPRS International Journal of Geo-Information*, 6(7), 206.
12. Zelinsky, A., Jarvis, R. A., Byrne, J. C., Yuta, S., (1993), Planning paths of complete coverage of an unstructured environment by a mobile robot, *Proceedings of the 3rd International Conference on Intelligent Autonomous Systems (IAS-3)*, 533-538.
13. Akpinar, B., (2021), Performance of different SLAM algorithms for indoor and outdoor mapping applications, *Applied System Innovation*, 4(4), 101.
14. Github, Github HDL Graph SLAM. https://github.com/koide3/hdl_graph_slam.git
15. Koide, K., Miura, J., Menegatti, E., (2019), A portable three-dimensional LIDAR-based system for long-term and wide-area people behavior measurement, *International Journal of Advanced Robotic Systems*, 16(2), 1729881419841532.
16. Arshad, S., Kim, G-W., (2021), Role of deep learning in loop closure detection for visual and lidar SLAM: A survey, *Sensors*, 21(4), 1243.
17. Xiong, B., Jiang, W., Li, D., Qi, M., (2021), Voxel grid-based fast registration of terrestrial point cloud, *Remote Sensing*, 13(10), 1905.



Goo Jun Ji

M.S. candidate in the Department of Mechanical Engineering, Hanyang University. His research interest is autonomous driving engineering.
E-mail: jiejg@kitech.re.kr



Myeong Gyu Lee

Ph.D. candidate in the Department of Mechanical Engineering, KOREA University of Technology and Education. His research interest is autonomous driving engineering.
E-mail: cobby6010@kitech.re.kr

**Won Gun Kim**

Dr. Kim is a principal researcher at Korea Institute of Industrial Technologies (KITECH). Ph.D. degrees in graduate school of automotive engineering from Seoul National University, Korea in 2012. His research interests are electric and autonomous vehicle, integrated driving.
E-mail: wgk@kitech.re.kr