한국정보전자통신기술학회논문지(jkiiect)'20-10, Vol.13 No.5
논문 20-13-05-377 http://dx.doi.org/10.17661/jkiiect.2020.13.5.377

LRF 센서를 이용한 글로벌 맵 기반의 적응형 이동 장애물 회피 알고리즘 개발

오세권*, 이유상**, 이대현***, 김영성****

Development of Adaptive Moving Obstacle Avoidance Algorithm Based on Global Map using LRF sensor

Se-Kwon Oh*, You-Sang Lee**, Dae-Hyun Lee***, Young-Sung Kim****

요 약 본 논문에서는 고정된 장애물이 포함된 글로벌 맵 환경에서 LRF 센서만을 가진 자율이동 로봇이 이동장애물을 회피하기 위한 알고리즘을 제안한다. 우선 이동장애물을 회피하기 위해 LRF 거리 센서 데이터와 글로벌 맵을 이용하여 이동장애물을 추출한다. 추출된 이동장애물과 자율이동 로봇의 상대적인 벡터 성분의 합을 이용해 타원 형태의 안전반경을 생성한다. 생성된 안전반경을 고려하여 자율이동 로봇이 이동장애물을 회피하고 목적지에 도착할 수 있도록 한다. 제안된 알고리즘을 검증하기 위해 정량적인 분석 방법을 사용하여 기존 알고리즘과 비교하고 분석한다. 분석 방법은 이동장애물이 없을 때를 기준으로 제안된 알고리즘과 기존의 알고리즘의 경로의 길이와 주행 시간을 비교한다. 제안된 알고리즘은 이동장애물의 상대적 속도와 방향을 고려하여 회피할 수 있어서 경로와 주행 시간 모두 기존의 알고리즘보다 높은 성능을 보인다.

Abstract In this paper, the autonomous mobile robot whit only LRF sensors proposes an algorithm for avoiding moving obstacles in an environment where a global map containing fixed obstacles. First of all, in oder to avoid moving obstacles, moving obstacles are extracted using LRF distance sensor data and a global map. An ellipse-shaped safety radius is created using the sum of relative vector components between the extracted moving obstacles and of the autonomuos mobile robot. Considering the created safety radius, the autonomous mobile robot can avoid moving obstacles and reach the destination. To verify the proposed algorithm, use quantitative analysis methods to compare and analyze with existing algorithms. The analysis method compares the length and run time of the proposed algorithm with the length of the path of the existing algorithm based on the absence of a moving obstacle. The proposed algorithm can be avoided by taking into account the relative speed and direction of the moving obstacle, so both the route and the driving time show higher performance than the existing algorithm.

Key Words : Autonomous, LRF Sensor, Moving Obstacle, Mobile Robot, Navigation, VFH

1. 서론

딥러닝 기술의 발전이 산업에 많은 영향을 미치고 있다. 그중에서도 자율 주행 자동차의 기술 발전이 두각을 나타내고 있다. 딥러닝을 이용해 사물이나, 사람을 인지하거나 차선을 판단해 주행할 수 있는 레벨 3등급의 자율 주행 차량이 양산하여 실생활에 사용하고 있다. 위에 언급했듯이 딥러닝 기술은 오브젝트를 효과적으로 인식하고 추적하는 방법에 사용하고 있다. 하지만 추적한 오브젝트를

* L-SAM AAM System 2 Team, LIG Nex1
Received September 10, 2020 Revised October 12, 2020 Accepted October 12, 2020

효율적으로 회피하는 제어 방법에는 많은 연구가 필요하다. 회피를 위해서는 인식된 오브젝트의 위치와 이동 방향, 속도의 정보가 필요하다. 이러한 정보는 영상 센서로 알 수 없다. 따라서 회피를 위해 거리 센서가 필요하다. 거리 센서는 적외선, 초음파, 레이저 등 다양한 종류의 거리 센서가 존재한다. 그중 적외선과 초음파는 반사와 산란에 의한 왜곡이 발생하기 때문에 먼 거리를 측정하기에는 불가능하다. 하지만 레이저 센서는 직진성이 좋아 사용에 적합하다. 그에 따라 현재 자율주행 차량에 LiDAR 센서를 사용하고 있다. LiDAR 센서는 도플러 측정을 통해 속도를 알 수 있는 Doppler lidar, 영상을 습득할 수 있는 Imaging lidar, 단순한 거리를 측정할 수 있는 range finder lidar 등이 있다. 본 논문에서는 LRF(Laser Range Finder)센서를 이용한다. LRF 센서는 센싱 주기가 빠르고 데이터의 양이 적어 가용성이 높아 낮은 시스템 환경에서도 높은 성능을 만족할 수 있으며 실내 환경에 사용에 적합하다.

따라서 본 논문에서는 거리를 측정할 수 있는 LRF 센서만을 가진 자율이동 로봇 이용해 이동하는 장애물을 효율적으로 회피할 수 있는 알고리즘을 개발하고, 실험을 통하여 검증을 수행한다. 우선 고정 장애물은 글로벌 맵을 통해 주워지며, 이동장애물은 LRF 센서로 인식하여 특성을 파악하고 모델링[1]을 수행한다. 이러한 기술은 기존의 KFBS(Kalman Filter Based Segmentation)[2] 알고리즘을 사용한다. KFBS알고리즘을 통해 세그먼트 된 센서 값의 변위 차를 이용하여 이동장애물을 추출한다. 다음으로 추출한 이동장애물을 회피하기 위해 기존 연구의 VFH(Vector Field Histogram)[3,4]의 알고리즘을 기반으로 개발을 수행한다. 이와 같이 VFH 기반의 이동장애물을 회피하기 위한 기존 논문의 EVFH(Enhanced Vector Field Histogram)[5]이 있다. 하지만 EVFH는 이동장애물과 자율이동 로봇의 상대적인 속도나 방향에 제약이 많고 충돌에 대한 문제도 존재한다. 이에 대한 문제점을 인식하고 이를 개선하기 위해 본 논문에서는 A-VFH(Adaptive Vector Field Histogram) 알고리즘을 개발하고 제안한다. 제안한 A-VFH 알고리즘은 기존 연구 EVFH의 검증 방법과 동일하게 수행하여 정량적으로 비교하고 성능을 입증한다.

본 논문 구성은 다음과 같다. 본론으로 2장에서 자율이동 로봇의 시스템을 기술하고 동적 환경에 대한 정의를 내린다. 3장에서는 이동장애물 탐지 알고리즘의 동작 순서대로 기술하며 4장에서는 기존 알고리즘에 대한 설명과 더불어 제안된 알고리즘을 상세히 기술한다. 다음으로 5장에서 제안된 알고리즘을 정량적인 방법을 통해 검증하고 성능을 입증한다. 마지막으로 6장에서 결론을 기술한다.

2. 자율이동 로봇의 시스템 구성

본 논문의 자율이동 로봇은 그림 1과 같이 센서 모듈(Sensor Module), 브레인 모듈(Brain Module), 모빌리티 모듈 (Mobility Module) 등으로 분리된다.[6]

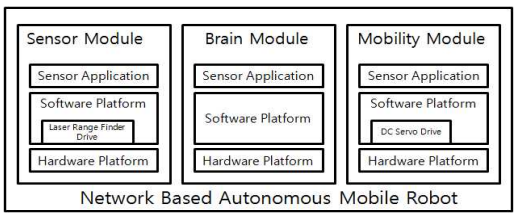

그림 1. 자율이동 로봇 구성도
Fig. 1. Block diagram of autonomous mobile robot

센서 모듈은 환경 정보 취득을 위한 센서를 제어하는 모듈로서, 전방 180도 46점 분해능을 가지는 LRF(Laser Range Finder)가 부착되어 있으며, 모바일 모듈은 로봇의 구동을 제어하는 모듈로서, 3개의 바퀴로 구성된 단방향 구동(Omni-Direction) 방식을 사용하고, 구동부의 엔코더(Encoder)를 이용하여 로봇의 현재 위치 정보인 오도메트리 정보를 계산하여 제공하는 역할을 한다. 브레인 모듈은 위치 판단의 확률적 계산 같은 복잡한 계산을 수행하는 역할을 한다.

본 논문에서의 환경은 글로벌 맵을 통해 벽이나 고정 장애물을 알 수 있다. 글로벌 맵은 SLAM 알고리즘[7]이나 맵 데이터를 통해 습득할 수 있다. 이는 본 논

문에서는 언급하지 않는다. 이동장애물은 글로벌 맵에 포함되지 않는 이동하는 장애물을 말한다. 또한, 이동장애물은 크기 및 속도 정보 등, 이동장애물의 모든 정보는 주어지지 않는다.

3. 이동장애물 탐지 및 모델링

3.1 LRF 센서 데이터 세그먼테이션

이동장애물을 탐지하기 위해서는 가장 먼저 장애물들 사이의 경계점을 찾아 LRF 센서 데이터를 세그먼트로 분리한다. 분리된 세그먼트는 LRF 센서 데이터를 오브젝트 형태의 장애물로 분리할 수 있다. 본 논문에서 세그먼테이션 수행을 위해 기존의 KFBS 알고리즘을 사용한다. KFBS 알고리즘은 칼만 필터를 이용하여 장애물의 경계점을 찾아 세그먼테이션을 하는 알고리즘이다.

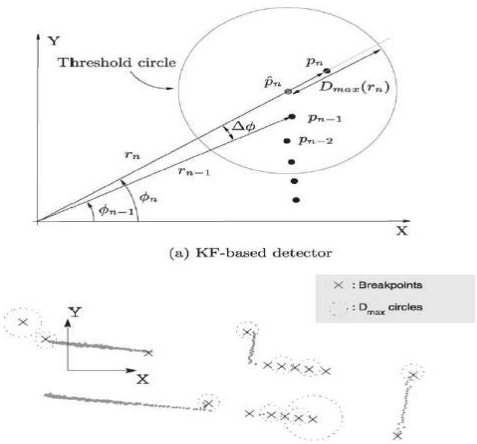

그림 2. KFBS 알고리즘
Fig. 2. KFBS algorithm

그림 2와 같이 한 주기의 n개의 센서 데이터 r_n이 있다. p_n은 r_n의 좌표 값이다. 만약 같은 하나의 장애물에 연속되는 센서 데이터 r_i, r_{i+1}가 측정될 경우, p_i, p_{i+1}값은 유사한 값을 갖게 되고 연속되는 센서 데이터 r_i, r_{i+1}가 다른 장애물에 있을 경우, p_i, p_{i+1}값의 차가 크게 발생한다. 이러한 원리를 통해 연속되는 센서 데이터에 칼만 필터를 적용하여 다음 센서 데이터의 좌표 값을 예측하여 만약 예측한 좌표 값 $\hat{p_n}$이 실제 센서 데이터의 좌표 값 p_n과의 차가 $D_{\max}$보다 크게 되면 다른 장애물로 인식하여 장애물의 경계점을 탐지한다.

3.2 이동장애물 추출 알고리즘

이동장애물 추출하는 문제를 해결하기 위해 글로벌 맵을 이용한다. 글로벌 맵은 고정 장애물에 대한 모든 정보를 갖고 있다. 이를 이용하여 그림 3과 같이 세그먼트 된 거리 센서 데이터에서 고정 장애물에 대한 정보를 가진 글로벌 맵의 정보를 빼면 이동장애물에 대한 정보만 남게 된다. 이를 통해 이동장애물만을 추출할 수 있다.

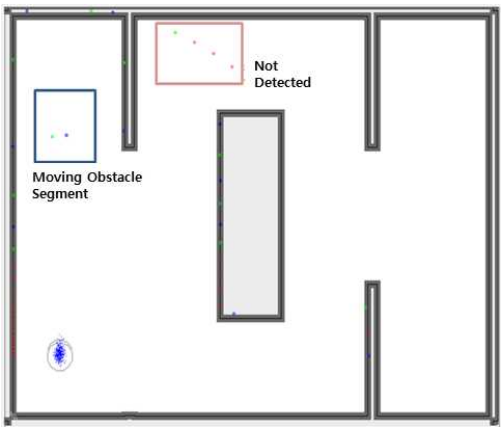

그림 3. 이동장애물 추출
Fig. 3. Extraction of moving obstacle

이동장애물 추출 알고리즘은 다음과 같다. 첫 번째, 글로벌 맵을 이용해 자율이동 로봇의 위치로부터 고정 장애물에 대한 위치 정보를 얻는다. 두 번째로 세그먼트된 n개의 센서 데이터가 고정 장애물의 위치 범위에 포함하는지 확인한다. 마지막으로 세그먼트된 n개의 센서 데이터 중 80% 이상이 고정 장애물 위치에 포함되면 고정 장애물로 판단하고 이보다 작게 되면 이동장애물로 판단하여 추출하게 된다. 예외 경우로, 센서 데이터의 거리 값이 센서의 최대 거리와 같다면 장애

물이 존재하지 않는 위치(그림 3의 Not Detected)이므로 그때의 세그먼트는 무시한다.

추출된 이동장애물 세그먼트들의 특성을 파악할 수 있도록 정확하게 모델링이 이뤄져야 한다. 하지만 거리 센서 데이터를 이용하여 모델링을 하는데 제약 조건이 따른다. 거리 센서로는 이동장애물의 뒷부분을 알 수 없으므로 로봇과 평행하고 기다란 형태의 이동장애물을 모델링하기 어렵다. 따라서 본 논문에서는 이동장애물은 이동하는 타원 형태의 장애물로 정의한다.

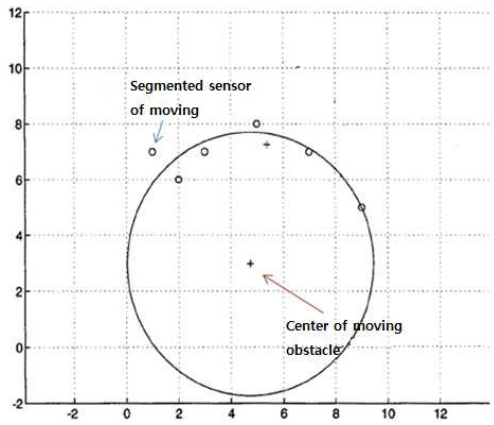

그림 4. LSCF(Least-Squares Circle Fit) 알고리즘
Fig. 4. LSCF(Least-Squares Circle Fit) algorithm

이동장애물 모델링은 이동장애물의 세그먼트를 특징점 추출 알고리즘을 통하여 이동장애물의 중점, 속력, 이동 방향, 크기를 알 수 있다. 특징점 추출 알고리즘은 LSCF(Least-Squares Circle Fit)[8] 알고리즘을 사용한다. LSCF 알고리즘은 선형 대수학의 회귀분석법을 사용한다. 그림 4와 같이 LSCF 알고리즘에서 독립변수는 세그먼트 된 센서 데이터이고 추출된 원의 중심점과 반지름은 종속변수로 정의된다. 또한, 식 (1), (2)을 통해 추출된 원의 중심점을 이전의 중심점과의 차이를 계산하여 이동장애물의 진행 방향과 속력을 알 수 있다.

$$\Delta CenterPos = \frac{NowCenter.x - PreCenter.x}{NowCenter.y - PreCenter.y}$$
$$Direction(^\circ) = \tan^{-1}(\Delta CenterPos) \qquad (1)$$

$$\Delta CenterDis = Dis(NowCenter, PreCenter)(km)$$
$$Velocity(km/h) = \frac{\Delta CenterDis(km)}{Time(h)} \qquad (2)$$

4. 이동장애물 회피 알고리즘

4.1 기존 EVFH(Enhanced Vector Field Histogram) 알고리즘

EVFH 알고리즘은 VFH 알고리즘 기반의 이동장애물을 회피할 수 있도록 제안된 알고리즘이다. VFH 알고리즘은 대표적 반응제어(Reactive-control) 방식의 장애물 회피 알고리즘이다. VFH는 그림 5와 같이 로봇 중심으로 주변의 히스토그램 그리드 맵을 생성한다. 생성된 히스토그램 그리드 맵에 매 주기마다 거리 센서의 위치 좌표 값을 그리드 맵의 Active Window에 Certainty Values값으로 저장하여 업데이트한다.

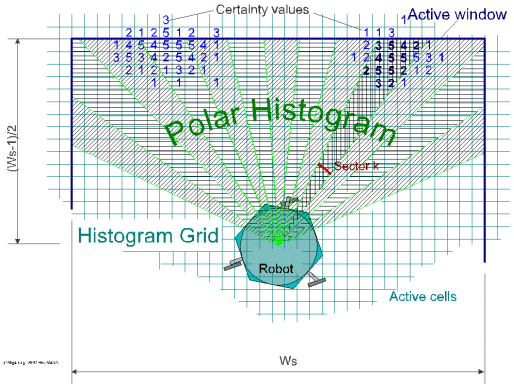

그림 5. VFH의 히스토그램 그리드 맵
Fig. 5. Histogram grid map of VFH

히스토그램 그리드 맵은 로봇 중심으로 360도 방향의 환경 정보를 갖는다. 하지만 로봇에 부착된 LRF 센서는 전방 180도만 탐지한다. 이러한 이유로 로봇의 뒤 180도 방향의 환경 정보를 저장하기 위해 이전 주기의 히스토그램 그리드 맵을 로봇이 이동하는 만큼 Certainty Values 값을 이동시켜 로봇 중심으로 360도의 환경 정보를 가질 수 있다. 360도의 환경 정보를 갖는 히스토그램 그리드 맵을 로봇 중심으로 전 방향에 존재하는 Certainty Values을 그림 6과 같이 폴라 히스토그램으로 변형을 한다.

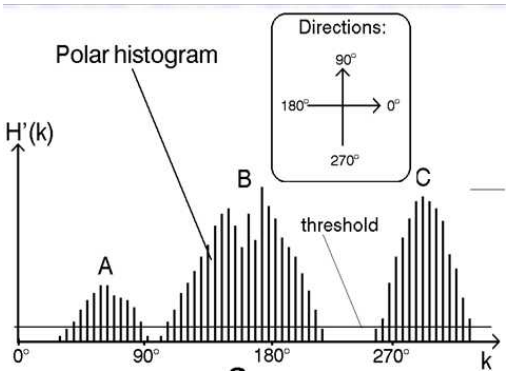

그림 6. VFH의 폴라 히스토그램
Fig. 6. Polar histogram of VFH

그림 6의 x축 섹터(Sector) k는 로봇 좌표에서의 로봇 방향에 대한 절대 각도를 나타내고, y축인 $H'(k)$는 장애물의 존재에 대한 확률값을 의미한다. 그러므로 $H'(k)$가 증가하면 장애물 존재 확률값도 같이 증가 된다. 반대로 $H'(k)$가 감소하면 장애물 존재 확률도 따라서 감소한다. 결과적으로 로봇은 VFH 알고리즘 결과의 $H'(k)$에서 설정한 임계값(Threshold)보다 낮은 섹터로 로봇은 이동하게 된다. 두 개 혹은 그 이상의 만족하는 결과 값이 나오면 목표 지점의 벡터와 현재 로봇의 진행 방향 벡터의 가중치에 의해 섹터를 선택하여 이동하게 된다.

하지만, VFH 알고리즘에는 이동장애물에 대한 정보가 포함되지 않아 이동장애물에 대한 회피를 보장할 수 없다. 따라서 EVFH 알고리즘을 사용하여 이동장애물을 회피할 수 있도록 한다. EVFH 알고리즘에서는 우선 이동장애물에 타원 형태의 안전반경을 생성해야 한다. 안전반경을 생성하는 이유는 이동장애물이 이동하는 특성이 있어서 이동장애물의 이동으로 로봇과 충돌 위험이 존재하기 때문이다. 이동장애물의 타원 안전반경을 생성하기 위해서 이전 3. 이동장애물 탐지 및 모델링을 통해 구한 데이터를 이용한다. 이동장애물의 타원 안전반경의 형태와 크기는 모델링을 통하여 구한 이동장애물의 현재의 중점좌표, 진행 방향, 속력, 크기에 의해 결정된다.

$$x = a \bullet \cos(r) - \lambda \tag{3}$$

$$y = b \bullet \sin(r) \tag{4}$$

타원 방정식은 식 (3), (4)과 같다. 이때 r은 0~360의 값을 갖고, a의 값은 타원의 장축에 해당하며, b값은 타원의 단축에 해당한다. λ의 값은 x축으로 평행이동 값이며, 이는 이동장애물의 중심이 타원의 중심보다 뒤쪽에 위치하도록 만들기 위해 적용하는 값이다.

$$a = k^{*}\, obstacle.velocity \tag{5}$$

$$b = k^{*}\, obstacle.size \tag{6}$$

이동장애물의 타원 안전반경의 장축과 단축은 식 (5), (6)과 같이 타원의 단축(b)은 안전반경의 크기와 관련되며 크기는 이동장애물의 크기 값에 의해 결정된다. 타원의 장축(a)은 이동장애물의 속력에 의해 결정된다. k의 값은 이동장애물의 형상에 따른 게인 값이며, 본 연구에서는 기존 연구와 같이 원 형태의 이동장애물로 정의하였기 때문에 기존 연구와 같이 $k=2$ 으로 정의한다.

타원의 장축의 방향은 이동장애물의 진행 방향으로 결정된다. 타원의 장축이 진행 방향으로 회전하기 위해서 다음 식 (7)을 이용하여 회전 이동한다.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{7}$$

x, y 의 값은 식 (3),(4)에 의해 생성된 타원의 좌표 값이며, x', y'는 회전 이동 때문에 결정된 타원의 좌표 값이다. θ값은 이동장애물의 진행 방향 값을 갖는다. 그림 7은 타원 안전반경을 적용한 이동장애물의 그림이다.

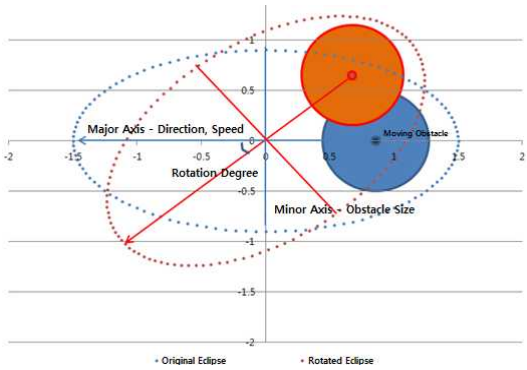

그림 7. 이동장애물의 타원 안전반경
Fig. 7. Elliptic safety radius of moving obstacle

4.2 A-VFH(Adaptive Vector Field Histogram) 알고리즘

이동장애물을 회피를 위한 EVFH 알고리즘은 이동장애물의 특성(속도, 방향)에 따라 안전반경을 생성하여 이동장애물을 회피하는 방법을 제안하였다. 하지만, 이동장애물과 자율이동 로봇의 상대적인 속도에 따라 회피 가능성을 보장할 수 없다. 예를 들어 이동장애물의 속도가 빠를 경우, 자율이동 로봇이 EVFH의 안전반경을 확인하고 피하려고 회전을 하는 동안 이동장애물이 접근하여 충돌이 발생할 수 있다. 또한, 이동장애물의 속도 대비 자율이동 로봇의 속도가 상대적으로 빠를 경우 자율이동 로봇이 이동장애물을 탐지하고 안전반경을 생성하기 전에 충돌이 발생할 수 있다. 이러한 문제점을 해결하기 위해 A-VFH 알고리즘을 제안한다.

4.2.1 벡터 내적을 이용한 타원 안전반경

기존 EVFH 알고리즘에서 이동장애물과 자율이동 로봇의 상대속도를 반영하지 못하는 문제를 해결을 위해 A-VFH 알고리즘에서는 VSCA(Vector Safety Circle Area)을 제안한다. VSCA는 이동장애물의 벡터와 자율이동 로봇의 벡터 성분의 합을 이용하여 이동장애물의 안전반경을 생성하는 방법이다. 우선, 자율이동 로봇과 이동장애물의 상대적 벡터는 이동방향과 속도를 통해 벡터를 생성한다. 생성된 벡터의 합을 통해 상대적 이동장애물의 경로를 예측할 수 있다. 따라서 그림 8과 같이 자율이동 로봇과 이동장애물이 움직일 때, 이동장애물의 이동 벡터와 자율이동 로봇의 보수 벡터의 합을 통해 상대적 이동장애물의 경로를 구한다.

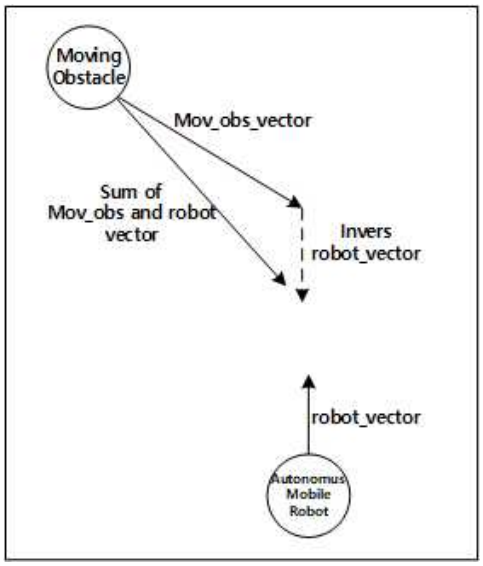

그림 8. 이동장애물과 자율이동 로봇의 벡터의 합
Fig. 8. Sum of moving obstacle and mobile robot vector

상대적 이동장애물의 경로를 통해 타원 안전반경을 생성하는데 그림 8과 같이 벡터의 합과 삼각형의 sin 법칙을 이용해 제안한 VSCA 타원 안전반경의 장축과 방향을 결정한다.

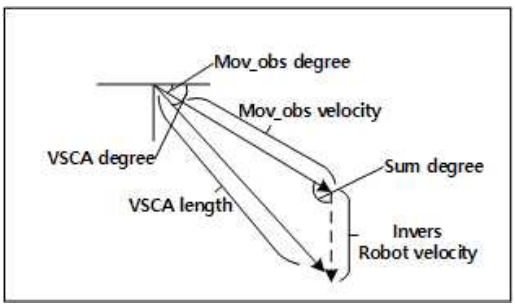

그림 9. VSCA의 장축과 방향
Fig. 9. Long-shaft and direction of VSCA

그림 9의 VSCA length는 타원 안전반경의 장축으로 정의하며, 이는 두 벡터의 합을 통해 생성된 벡터로 구한다. 따라서 이동장애물의 속도와 로봇의 속도 그리고 둘 사이의 각도를 이용하여 제2 코사인법칙 공식을 이용해 식 (8)과 같이 도출한다.

$$\begin{aligned} sum\deg &= Movobsdeg + robotdeg \\ m &= movosbvelocity \\ r &= -robotvelocity \\ VSCA\,length &= \sqrt{m^2 + r^2 - 2mr\cos(sum\deg)} \end{aligned} \quad (8)$$

VSCA degree는 VSCA 타원 안전반경의 방향으로 정의하며, 각도는 위와 같이 세 변의 길이를 알고 있으므로 제2 코사인법칙을 통해 식 (9)와 같이 제안한 VSCA 타원 안전반경의 진행 방향의 각을 구한다.

$$\begin{aligned} m &= movosbvelocity \\ r &= -robotvelocity \\ v &= VSCAlength \\ vectordeg &= \arccos\left(\frac{(m^2+v^2-r^2)}{2mv}\right) \\ VSCAdeg &= vectordeg + movobs \end{aligned} \quad (9)$$

4.2.2 이동 방향 적응형 Certainty Values

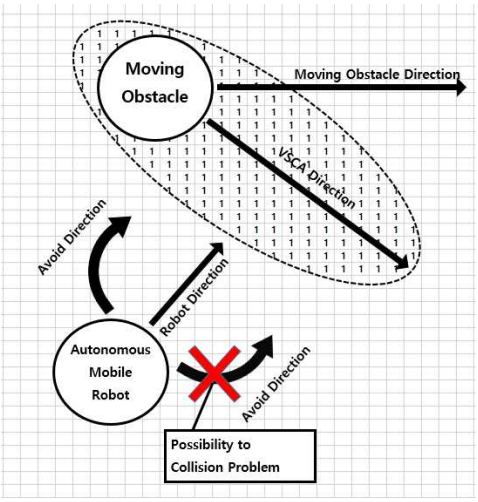

그림 10. 기존의 certainty value 생성 방법
Fig. 10. Existing method of creating certainty value

그림 10과 같이 A-VFH에 의한 타원 안전반경이 생성되었을 때 자율이동 로봇이 이동 방향을 결정함에 문제가 발생할 수 있다. 이동장애물의 방향과 자율이동 로봇의 회피 방향이 같을 때 충돌의 위험이 발생하거나 반복적 회피가 발생할 수 있다.

이를 해결하기 위해 그림 11과 같이 Certainty Values를 이동장애물의 방향으로 증분을 시켜 반영한다. 이에 따라, Certainty Values가 이동장애물의 진행 방향과 반대 방향으로 최소의 값을 갖게 된다. 최소 히스토그램을 갖는 방향으로 이동하는 A-VFH 알고리즘을 통해 이동장애물의 이동 방향과 반대 방향으로 회피하여 이동장애물과 충돌을 막을 수 있으며, 자율이동 로봇의 이동 거리를 최소화할 수 있다.

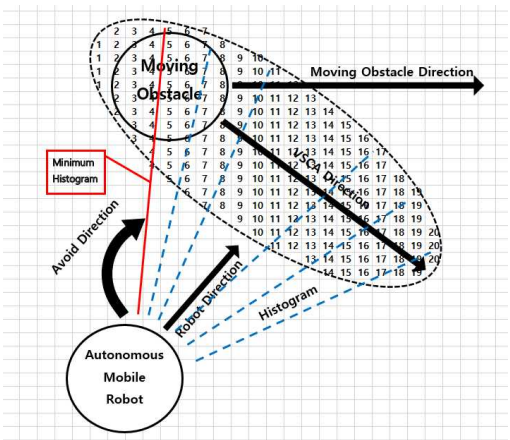

그림 11. VSCA의 certainty value
Fig. 11. Certainty value of VSCA

제안한 방법으로 생성된 이동장애물에 대한 VSCA 타원 안전반경의 Certainty Values를 고정장애물을 포함한 히스토그램 그리드 맵에 주기마다 업데이트하여 최종 Certainty Values로 저장한다.

$$X_{histo} = ChangeHistoX(x' + obsCenter.x) \quad (10)$$

$$Y_{histo} = ChangeHistoY(y' + obsCenter.y) \quad (11)$$

A-VHF에서 도출된 x', y' 값을 식 (10), (11)과 같이 이동장애물의 위치에 히스토그램 그리드 맵 좌표로 변환하여 저장한다. $obsCenter$는 이동장애물의 좌표 값이다. $ChangeHistoX()$, $ChangeHistoY()$ 는 로봇 좌표를 히스토그램 그리드 맵 좌표로 변환하는 함수이다. 위와 같이 업데이트 된 히스토그램 그리드 맵을 폴라 히스토그램으로 변경을 한다. 폴라 히스토그램을 통해 로봇의 주행 방향을 설정하는 방법은 기존의 VFH와 같이 폴라 히스토그램이 임계 값보다 적은 섹터 방향으로 주행 방향을 설정한다. 이처럼 A-VFH는 이동장애물에 대한 VSCA 타원 안전반경을 생성하고 그 정보를 히스토그램 그리드 맵에 업데이트함으로써 이동장애물의 회피 안전성을 보장할 수 있다.

5. 실험 결과

5.1 실험환경 구성

본 논문에서 제안한 이동장애물 탐지 및 회피 알고리즘 실험의 환경은 기존 논문[5]의 구성과 동일한 방법으로 구성한다.

5.2 A-VFH 알고리즘 실험 검증

제안한 장애물 회피 알고리즘의 효율을 정량적으로 평가하기 위해 그림 12와 같이 이동 경로의 효율 평가[9]를 사용한다. 이동 경로의 효율 평가는 식 (12)와 같이 가시성 그래프에서 생성된 이동장애물이 없을 때의 최단 경로의 길이를 l로 놓고, 실제 이동 경로 사이의 편차 d_p를 최단 경로 l로 나눈 값의 평균으로 이루어진다. 따라서 이동 경로의 정량적 효율 평가 값이 작을수록 더 효율적인 경로가 생성됐음을 의미한다.

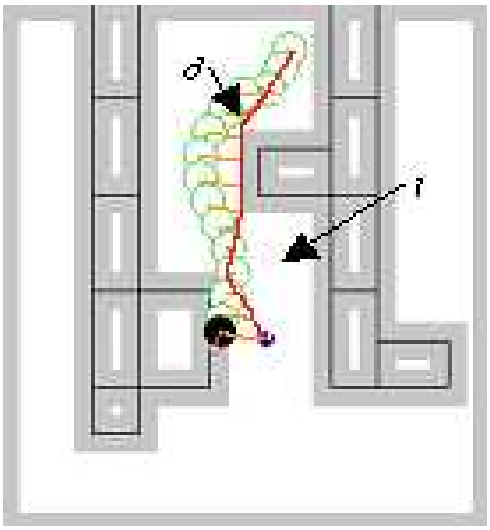

그림 12. 이동 경로 효율 정량적 평가 지표
Fig. 12. Efficient evaluation index of path

$$s = (\sum d_p) / l \tag{12}$$

A-VFH 알고리즘 검증은 2가지의 실험환경 CASE를 구성하여 검증을 진행한다. 첫 번째로는 단순한 동적 환경에서 이동장애물과 로봇의 진행 방향이 한 직선상에 있으며 진행 방향이 마주 보고 있을 때, 두 번째는 단순한 동적 환경에서 이동장애물과 로봇의 진행 방향이 수직일 때로 검증을 진행한다. 실험환경 CASE에서는 이동장애물이 있을 때 기존의 EVFH 알고리즘을 사용한 경우, 제안한 A-VFH 알고리즘을 사용한 경우로 나눠 실험한다. 그리고 이동장애물과 로봇의 진행 방향이 수직일 때는 이동장애물의 속력을 변경하여 이동장애물 회피 알고리즘의 정량적 평가와 이동장애물과 로봇의 충돌에 대한 안정성을 검증하고자 한다.

5.2.1 로봇과 이동장애물이 직선으로 움직일 경우

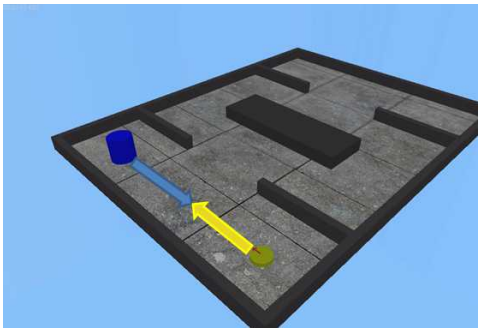

그림 13. 실험1 구성도
Fig. 13. Configuration diagram of experiment 1

그림 13은 첫 번째 실험으로 이동장애물과 자율이동 로봇이 한 직선상에 있으며, 서로 마주보는 방향으로 직선 교차할 때의 구성이다. 해당 실험 조건에서 2가지 속도 조건 0.416m/s 와 0.614m/s로 수행하여 결과를 분석한다.

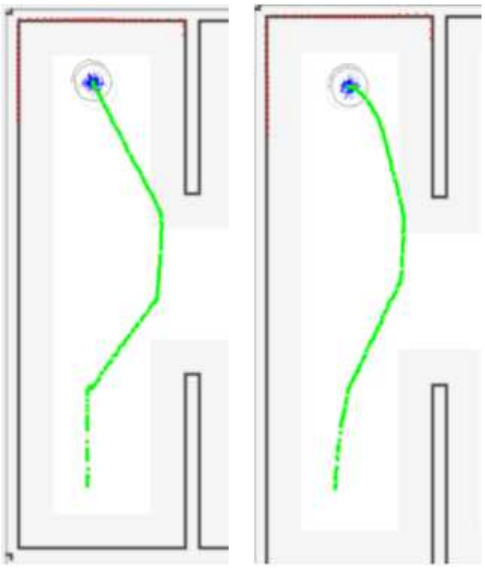

그림 14. 이동장애물 속도 : 0.416m/s일 때
좌) EVFH 우) A-VFH
Fig. 14. Moving obstacle speed : at 0.416m/s
Left) EVFH Right) A-VFH

표 1. 이동장애물 속도 : 0.416m/s 일 때
Table 1. Moving obstacle speed : at 0.416m/s

Spec.	EVFH	A-VFH
Path efficiency (s)	1.2150	1.2010
Path length (l')	5.441 m	5.311 m
Driving time (t')	23.100 s	18.70s
Average Speed (v')	0.236 m/s	0.284 m/s

그림 14는 속도 0.416m/s에서 EVFH와 A-VFH 알고리즘을 수행한 결과이다. EVFH 와 A-VFH 모두 성공적으로 이동장애물을 회피할 수 있었다. 하지만 A-VFH는 타원 안전반경의 장축의 길이가 상대적 벡터의 합으로 생성되므로 장축의 길이가 길어 EVFH보다 먼저 이동장애물을 인식하고 회피를 미리 수행하였다. 따라서 자율이동 로봇의 평균 속도가 빠르고 목표 지점에 도착하는 시간이 단축되었다. 표 1은 수행한 결과를 정량적 지표로 표현한 것이며, 기존 EVFH 알고리즘 보다 제안한 A-VFH의 효율성이 높은 것을 알 수 있다.

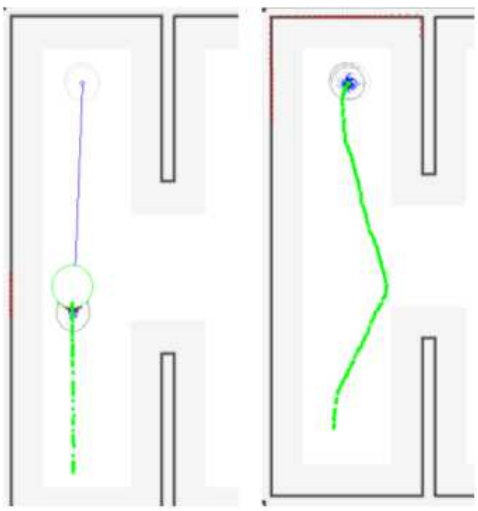

그림 15. 이동장애물 속도 : 0.614m/s일 때
좌) EVFH(충돌) 우) A-VFH
Fig. 15. Moving obstacle speed : at 0.614m/s
Left) EVFH(Collision) Right) A-VFH

그림 15는 속도 0.614m/s에서 EVFH와 A-VFH 알고리즘을 수행한 결과이다. 이 경우에는 기존의 EVFH 알고리즘은 이동장애물의 타원 안전반경을 인식하고 회피 동작을 수행하는 동안 이동장애물이 빠르게 다가와 충돌이 발생하는 것을 확인할 수 있었다. 그에 반에 A-VFH 알고리즘은 EVFH 보다 더 빠르게 이동장애물의 타원 안전반경을 인식하고 회피할 수 있어서 충돌 없이 목표 지점에 도착할 수 있었다.

다음 표 2는 EVFH에서 이미 충돌이 발생하여 정량적으로 비교를 할 수 없으나 제안한 A-VFH 알고리즘은 회피에 성공하므로 기존의 알고리즘보다 향상된 알고리즘이란 것을 알 수 있었다.

표 2. 이동장애물 속도 : 0.614m/s 일때
Table 2. Moving obstacle speed : at 0.614m/s

Spec.	EVFH	A-VFH
Path efficiency (s)	collision	1.387
Path length (l')	collision	5.523
Driving time (t')	collision	23.12 s
Average Speed (v')	collision	0.201 m/s

5.2.2 로봇과 이동장애물이 수직으로 움직일 경우

그림 16은 두 번째 실험으로 이동장애물과 자율이동 로봇이 수직으로 교차할 때의 구성이다. 이 실험의 경우 이동장애물의 속도가 0.208m/s인 상황에서 실험을 수행하였다.

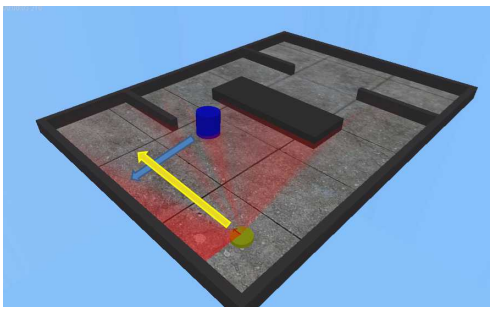

그림 16. 실험2 구성도
Fig. 16. Configuration diagram of experiment 2

그림 17은 수직으로 교차하고 이동장애물이 0.416m/s 속도를 가질 때 기존의 EVFH 알고리즘을 수행한 결과이다. EVFH 알고리즘은 그림 17 좌)의 결과처럼 정상적으로 회피를 수행할 수 있었다. 하지만, 그림 17 우)의 경우처럼 회피하지 못하고 충돌이 발생하는 경우가 발생한다. 이는 이동장애물의 타원 안전반경을 인식하였지만 동등한 Certainty Values를 갖고 있어 자율이동 로봇의 회피 방향이 이동장애물의 진행 방향과 같게 되면 충돌이 발생할 수 있다.

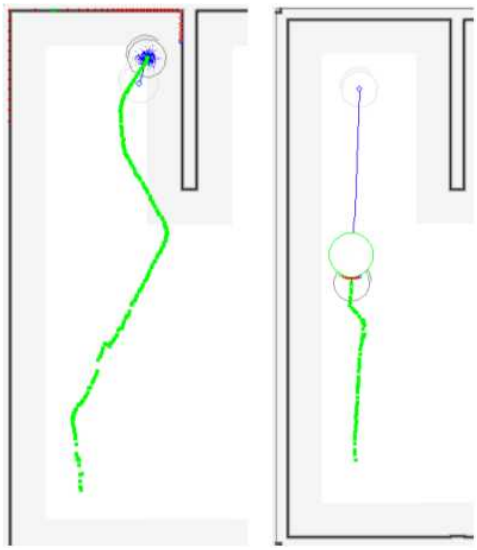

그림 17. 이동장애물 속도 : 0.416m/s일 때
좌) EVFH(성공) 우) EVFH (충돌)
Fig. 17. Moving obstacle speed : at 0.416m/s
Left) EVFH(Success) Right) EVFH (Collision)

위 그림 17 우)와 같이 EVFH 알고리즘과 같은 문제를 인식하고 A-VFH 알고리즘에서 3.2.2의 내용과 같이 Certainity Values를 이동장애물의 이동 방향으로 증분 하여 이동장애물의 이동 방향과 반대 방향으로 회피할 수 있도록 하였다. 따라서 기존의 EVFH 알고리즘과 달리 반복수행을 통해 충돌 영향성에 대한 검토를 수행하였고, 충돌이 발생할 가능성이 기존 알고리즘보다 적음을 알 수 있었다. 그림 18은 A-VFH 알고리즘을 수행한 결과이다. 기존 EVFH 알고리즘

이 회피 성공한 경우보다 이동장애물에 대한 VSCA 타원 안전반경으로 빠르게 회피 경로를 인지하고 동작하여 경로 효율 높아지고 그에 따라, 알고리즘의 성능이 향상됨을 알 수 있었다. 또한, 표 3의 정량적 지표에서도 제안한 A-VFH 알고리즘의 경로 효율성이 EVFH 알고리즘보다 높은 것을 알 수 있었다.

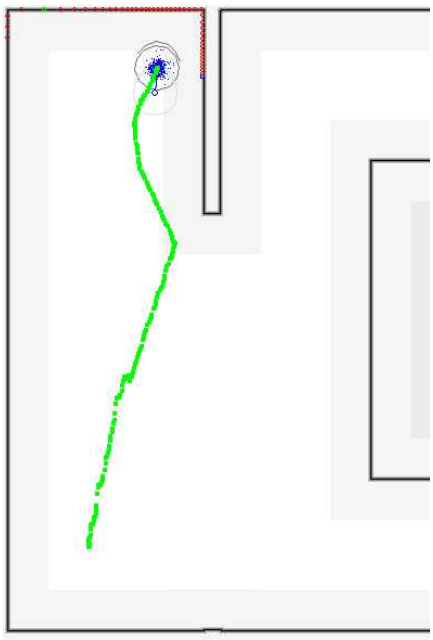

그림 18. 이동장애물 속도 : 0.416m/s일 때 A-VFH
Fig. 18. Moving obstacle speed : at 0.416m/s A-VFH

표 3. 이동장애물 속도 : 0.416m/s 일 때
Table 3. Moving obstacle speed : at 0.416m/s

Spec.	EVFH		A-VFH
-	Test1	Test 2	-
Path efficiency (s)	1.1484	collision	1.1009
Path length (l')	6.051 m	collision	5.732 m
Driving time (t')	25.700 s	collision	22.650 s
Average Speed (v')	0.235 m/s	collision	0.253 m/s

6. 결론

본 논문에서는 고정 장애물이 존재하는 환경에서 LRF 센서로 이용해 탐지된 이동장애물을 회피하는 알고리즘을 제안하였다. 기존 EVFH 알고리즘은 이동장애물이 속도가 빠르거나, 자율이동 로봇과 이동장애물의 상대적 방향에 따른 충돌 문제가 있다. 이러한 회피 문제점을 해결하기 위해 A-VFH 알고리즘을 제안하였다. 제안된 알고리즘의 실험 결과를 정량적으로 측정하기 위해 경로 효율, 경로 길이, 주행 시간을 측정하였다. 첫 번째 실험인 이동 방향이 같은 직선상에 있고 이동장애물의 속도가 0.416m/s일 경우 제안된 알고리즘이 경로 효율에 월등히 높은 성능을 만족하지 못했다. 하지만 이동장애물을 미리 인식하고 회피를 수행할 수 있어 자율이동 로봇의 평균 속도가 높아 주행 시간이 기존 알고리즘보다 19.04%가 단축되었다. 또한, 이동장애물의 속도가 0.614m/s일 때 기존 알고리즘에서 회피할 수 없었지만 제안된 알고리즘에서는 회피할 수 있었다. 두 번째로 이동 방향이 수직일 경우에도 기존 알고리즘에서 회피가 상황에 따라 불가능하였지만 제안된 알고리즘에서는 회피할 수 있었다. 이러한 결과로 볼 때 복잡한 구조에서도 제안된 알고리즘이 높은 성능을 보일 수 있을 것으로 판단된다.

본 논문에서는 LRF 센서만을 이용하여 이동장애물을 탐지하고 모델링을 수행하는 기존의 방법을 이용하였다. 하지만, LRF 센서만을 이용한 이동장애물 모델링은 한계가 있다. 따라서 향후 연구과제로 카메라 센서를 이용하여 현재 성숙도가 높아지고 있는 딥러닝 기술을 적용한 이동장애물 탐지 및 모델링을 수행하는 연구가 필요하다.

REFERENCES

[1] Se-Kwon Oh, Joo-Min Kim, Dae-Won Kim, “Development of a Moving Obstacle Detection Algorithm using a LRF Sensor in Dynamic Environments”, KIEE, 1385-1386, 2012.7

[2] GEOVANY ARAUJO BORGES, MARIE ALDON, "Line Extraction 2D Range Images for Mobile Robotics", Journal of Intelligent and Robotics Systems, 40, 267-297, 2004

[3] J. Borenstein and Y. Koren, "The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robot," IEEE Journal of Robotics and Automation, Vol. 7 No. 3, pp.278-288, 1991.

[4] J. Borenstein and Y. Koren, "Real-Time Obstacle Avoidance for Fast Mobile Robots," IEEE Transaction on Systems, Man, and Cybernetics, Vol. 19, No. 5, pp.1179-1187, 1989.

[5] Se-Kwon Oh, “Development of a Global Map-Based Moving Obstacle Detection and Avoidance Algorithm using a LRF Sensor in Dynamic Environments”, Myoungji University, 16-25, 2013

[6] Hongryeol Kim, Joomin Kim, and Daewon Kim, "Development of Coordinated Scheduling Strategy with Network Response Time Analysis for the CAN-Based Distributed Control systems," Proceeding of 2004 IEEE/RSJ International Conference on Intelligent Robot and Systems, pp.2099-2104, 2004.

[7] J. Vandorpe, H. Van Brussel, H. Xu," Exact Dynamic Map Building for a Mobile Robot using Geometrical Primitives Produced by a 2D Range Finder",

[8] WALTER GANDER, GENE H. GOLUB and ROLF STREBEL, "LEAST-SQUARE FITTING OF CIRCLES AND ELLIPSES", BIT, 40, 558-578, 1994

[9] Jin-Woo Kim, “Development of an Efficient Obstacle Avoidance Comprensation Algorithm Considering a Network Delay for a Network-based Autonomous Mobile Robot”, Myoungji University, 21-31, 2011

저자약력

오 세 권 (Se-Kwon Oh)

2011 명지대학교 정보통신공학과
2013 명지대학교 정보통신공학과
2013 ~ 현재 LIG넥스원 선임연구원

〈관심분야〉 유도무기, 탄 체계, 체계공학, 소프트웨어공학

이 유 상 (You-Sang Lee)

2006 경희대학교 컴퓨터공학과
2006 ~ 현재 LIG넥스원 선임연구원

〈관심분야〉 유도무기, 정보통신

이 대 현 (Dae-Hyun Lee)

2008 경희대학교 동서의료공학과
2010 경희대학교 동서의료공학과
2013 ~ 현재 LIG넥스원 선임연구원

〈관심분야〉 유도무기, 탄 체계, 신호처리, 임베디드 시스템

김 영 성 (Young-Sung Kim)

2015 한양대학교 에리카캠퍼스 컴퓨터공학과 학사
2017 한양대학교 컴퓨터공학과 석사
2017 ~ 현재 LIG넥스원 선임연구원

〈관심분야〉 유도무기, 탄 체계, 신호처리, 임베디드 시스템