

# Virtual Tangential Vector와 퍼지 제어를 이용한 준 3차원 경로계획

## Semi-3D Path Planning using Virtual Tangential Vector and Fuzzy Control

곽 경 운<sup>1</sup>, 정 해 관<sup>2</sup>, 김 수 현<sup>3</sup>

Kyung Woon Kwak<sup>1</sup>, Hae Kwan Jeong<sup>2</sup>, Soo Hyun Kim<sup>3</sup>

**Abstract** In this paper, a hybrid semi-3D path planning algorithm combining Virtual Tangential Vector(VTV) and fuzzy control is proposed. 3D dynamic environmental factors are reflected to the 2D path planning model, VTV. As a result, the robot can control direction from 2D path planning algorithm VTV and speed as well depending on the fuzzy inputs such as the distance between the robot and obstacle, roughness and slope. Performances and feasibilities of the suggested method are demonstrated by using Matlab simulations. Simulation results show that fuzzy rules and obstacle avoidance methods are working properly toward virtual 3D environments. The proposed hybrid semi-3D path planning is expected to be well applicable to a real life environment, considering its simplicity and realistic nature of the dynamic factors included.

**Keywords** : Gravity Vector, Tangential Vector, Fuzzy Control, Encirclement, Enlargement, Safety

### 1. 서론

주행로봇의 자율주행기술은 크게 위치인식(Localization), 지도생성(Mapping), 경로계획(Path planning) 기술로 나뉘고, Localization과 Mapping을 합쳐 보통 SLAM (Simultaneous Localization and Mapping)이라고 한다. 이 세 가지 기술 중에 경로계획 기술은 로봇이 장애물을 회피하면서 목표점(Local Goal)까지 최적의 경로로 도달하게 하는 중요한 기술이라 할 수 있다. 경로계획은 전역 경로계획과 지역 경로계획으로 구분하는데 전역 경로계획이란 이미 로봇이 이동할 경로의 정보를 어느 정도 알고 있을 때 자율적으로 경로를 생성하는 것이고, 지역 경로계획이란 전역 경로를 추종하면서 지도 정보로부터 나

와 있지 않은 예기치 못한 장애물이나 움직이는 장애물에 대해 여러 가지 센서를 이용하여 장애물을 피해가며 경로를 재생성하는 것이다.

정찰 주행로봇은 주어진 전역 지도를 기반으로 전장에 침투하여 알 수 없었던 유용한 정보를 아군에게 제공하는 역할을 한다. 그러나 전역 지도에서 제공하지 않은 예기치 못한 장애물이 존재할 수 있기에 로봇은 센서를 이용하여 새로운 경로를 탐색하는 지역 경로계획을 수행해야 한다. 지금까지 지역 경로계획과 장애물 회피에 대한 알고리즘이 꾸준히 연구가 진행되었다. Virtual Force Field (VFF)<sup>[1]</sup>은 포텐셜 필드 방법을 이용하였으며 기본 개념은 가상의 인력을 목표점으로부터 발생시키고 장애물로부터는 가상의 척력을 작용시켜 합력 방향으로 이동하게 하는 것이다. 이 방법은 간단하면서도 구현하기가 쉽지만 좁은 통로에서 발생하는 지그재그 현상이나 지역 국소점(Local Minima)에 빠지는 단점을 가지고 있다. Vector Field Histogram(VFH)<sup>[2]</sup>은 VFF를 보완하기 위한 방법으로 로봇 주위의 방사형 거리를 추출하여 조향 영역에 대

Received: Feb. 26, 2010; Reviewed: May. 04, 2010; Accepted: May. 07, 2010

<sup>1</sup> KAIST 기계항공시스템학부 기계공학전공 박사과정

(Email : kwakkw@kaist.ac.kr)

<sup>2</sup> 연구개발인력교육원 기획연수 1팀(Email : hothip29@kaist.ac.kr)

<sup>3</sup> KAIST 기계항공시스템학부 교수(Email : peaceall@kaist.ac.kr)

한 장애물 밀도를 Polar Histogram으로 도식화한 다음, 가장 낮은 장애물 밀도를 갖는 방향으로 회피하는 방법이다. 좁은 구역에서도 안정적으로 주행을 할 수 있지만 여전히 U자 형태의 장애물에 대한 지역 국소점 현상은 해결하지 못하였다. Dynamic Window Approach(DWA)<sup>[3]</sup>는 로봇의 동역학을 고려하는 속도 기반 장애물 회피 기법이다. DWA는 로봇의 선속도와 각속도를 각각 한 축으로 하는 속도 공간에서 운용 가능 속도의 집합인 Dynamic Window를 정의한 뒤 목표점으로서의 경로 생성, 현재 속도 유지, 장애물과의 여유 거리를 고려한 비용함수(Cost Function)에 대하여 최적의 속도를 추출한다. 따라서 동역학적으로 불가능한 경로를 배제하는 장점이 있는 반면 내부 변수에 대한 민감도가 알고리즘 성능에 직접적인 영향을 미친다는 단점이 있다. 한편 Nearness Diagram(ND)<sup>[4]</sup>은 'Divide and Conquer' 개념을 기본으로 하는 규칙 기반 알고리즘으로, 로봇이 주행 중 겪을 수 있는 모든 상황을 총 4개의 판단 기준과 5개의 의사 결정만으로 대처한다. 따라서 로봇은 어떠한 상황에서도 명확하게 의사 결정을 내릴 수 있으므로 매우 협소한 구역에 대해서도 좋은 성능을 낼 수 있다. 그러나 로봇의 기구학이나 동역학을 고려하지 않으므로 전 방향 조향 기반의 원형 로봇 외에는 적용하기 힘들다는 단점이 있다. 선행 연구 결과를 정리한 내용은 표 1과 같다<sup>[5]</sup>.

앞에서 언급했듯이 경로계획은 많은 응용분야가 있고 우수한 성능을 보이지만 U 형태의 장애물을 극복하지 못하는 지역 국소점 문제의 한계를 가지고 있다. 또한 대부분의 연구는 지면의 거칠기, 경사 등의 3D 요소들을 고려하지 않고 2D 환경에 한정되어 있다. 이에 본 논문에서는 지면의 거칠기, 경사와 같은 주행 환경의 3D 요소들까지 고려하여 로봇의 안전을 고려한 주행 속력과 경로까지 생성하는 방법을 제안한다. 이때 지면의 거칠기나

경사, 장애물로부터의 거리는 예측이 어렵고 애매성이 크기 때문에 로봇의 속력을 제어하고 산출하기 위해 퍼지 제어 이론<sup>[6, 7]</sup>을 이용하고 경로 방향을 생성하기 위한 방법으로는 Virtual Tangential Vector (VTV) 알고리즘<sup>[8]</sup>을 제안한다. Fuzzy Logic Control, Artificial Neural Networks-based Control, Genetic Algorithm의 3가지 기술을 묶어 소프트 컴퓨팅(Soft Computing) 기술이라 한다. 소프트 컴퓨팅은 비수치적이고 언어적인 기술로 본 논문에서는 앞서 언급한대로 로봇의 일반적인 구동에 대한 제어가 아닌 준 3차원 환경을 인식하여 속력자체를 제어하지 않고 퍼지 제어를 통해 출력으로 얻는 새로운 기법을 제안한다.

## 2. VTV와 퍼지 제어를 이용한 준 3차원 경로계획

주행로봇이 장애물과 충돌하지 않고 안전하게 목표점에 도달하기 위해 측정 데이터를 이용하여 실시간 장애물 회피가 가능한 VTV와 함께 경사나 지면 상태에 따라 속력 제어가 가능한 퍼지 제어를 제안한다.

### 2.1 2차원 경로계획

지역 경로계획을 위해 개발된 알고리즘은 VTV를 이용한 실시간 장애물 회피 기법이다. VTV는 LMS(Laser Measurement System)를 이용하여 감지된 장애물을 원형화, 확대한 후 장애물과의 충돌을 피하기 위해 목표점으로부터의 인력 벡터(Gravity Vector)와 장애물로부터의 접선 벡터(Tangential Vector)를 사용하여 지역 경로를 생성하는 것을 기본 개념으로 한다. VTV의 가장 큰 장점은 알고리즘 자체가 간단하면서도 지금까지 해결하지 못하였던 지역 국소점 현상을 부분적으로 해결하였다는 것이다.

표 1. 장애물 회피 방법에 대한 선행연구

Algorithm	Year	Developer	Concept
Virtual Force Field	1989	Johann Borenstein et al., University of Michigan, USA	<ul style="list-style-type: none"> <li>• Fusion of potential method and histogram grid</li> <li>• Virtually attractive and repulsive forces considered each for a goal and obstacle</li> </ul>
Vector Field Histogram	1991	Johann Borenstein et al., University of Michigan, USA	<ul style="list-style-type: none"> <li>• Extraction of candidate direction based on polar histogram generated from peripheral sensor data</li> </ul>
Dynamic Window Approach	1996	Dieter Fox et al., University of Bonn	<ul style="list-style-type: none"> <li>• Path planning based on both translational and rotational velocity to consider dynamics</li> <li>• Choice of best velocities by optimization of an objective function</li> </ul>
Nearness Diagram	2000	Javier Minguez et al., University of Zaragoza, Spain	<ul style="list-style-type: none"> <li>• 'Divide and Conquer' strategy</li> <li>• Only 5 situations and actions for 4 criterion</li> </ul>

### 2.1.1 장애물의 원형화와 확대

VTV는 계산량 감소와 구현 편의를 위하여 장애물을 원형화하고 확대시켜 가상의 장애물을 생성한다. 접선 벡터를 만들어 내기 위하여 장애물을 가상의 원으로 만들고, 알고리즘의 편의와 적용성을 위하여 원형화된 장애물을 로봇의 크기에 맞게 확대하는 방법을 제안한다. 실제 장애물을 가상의 원으로 대체하는 원형화 방법은 레이저 센서로부터 얻는 세 점으로부터 얻어낸다. LMS로 장애물로부터의 거리를 TOF(Time Of Flight) 원리에 의해 얻게 되고 LMS에 내장된 Step Motor로부터 장애물이 감지되는 방향(각도)을 얻는다. LMS로부터 장애물이 감지되는 양 끝점과 가장 가까운 점으로부터 세 점의 데이터를 받아 유일한 원을 얻을 수가 있는데 이를 장애물의 원형화라 하고 그림 1과 같다. 로봇이 진행함에 따라 실시간으로 세 점의 값들이 변하므로 여기서 얻어지는 원의 크기와 위치도 계속 변하게 된다. 또한 매 시간마다 원형화된 장애물은 주행로봇의 가장 긴 방향으로의 지름의 크기( $R_{robot}$ )만큼 확대시킨다. 이는 위에서 언급한 대로 계산

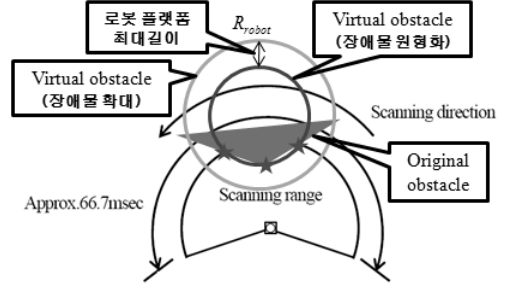


그림 1. 장애물의 원형화와 확대

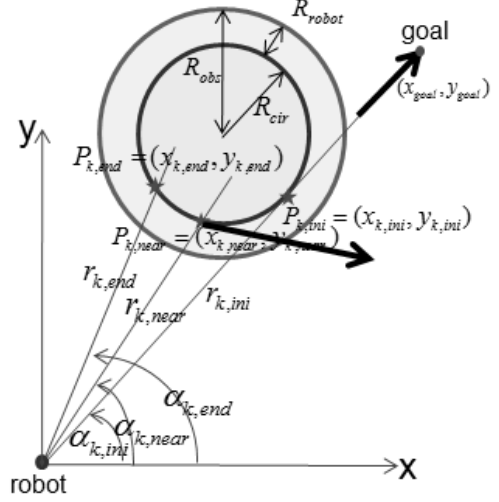


그림 2. VTV 구현을 위한 작용 벡터 및 변수

$$\begin{aligned}
 P_{k,ini} &= (x_{k,ini}, y_{k,ini}) \\
 P_{k,end} &= (x_{k,end}, y_{k,end}) \\
 P_{k,near} &= (x_{k,near}, y_{k,near}) \\
 x_{k,ini} &= r_{k,ini} \times \cos \alpha_{k,ini}, y_{k,ini} = r_{k,ini} \times \sin \alpha_{k,ini} \\
 x_{k,end} &= r_{k,end} \times \cos \alpha_{k,end}, y_{k,end} = r_{k,end} \times \sin \alpha_{k,end} \\
 x_{k,near} &= r_{k,near} \times \cos \alpha_{k,near}, y_{k,near} = r_{k,near} \times \sin \alpha_{k,near} \\
 y_a &= m_a(x - x_{k,ini}) + y_{k,ini} \\
 y_b &= m_b(x - x_{k,near}) + y_{k,near} \\
 m_a &= \frac{y_{k,near} - y_{k,ini}}{x_{k,near} - x_{k,ini}}, m_b = \frac{y_{k,end} - y_{k,near}}{x_{k,end} - x_{k,near}} \\
 y'_a &= -\frac{1}{m_a} \left( x - \frac{x_{k,ini} + x_{k,near}}{2} \right) + \frac{y_{k,ini} + y_{k,near}}{2} \\
 y'_b &= -\frac{1}{m_b} \left( x - \frac{x_{k,near} + x_{k,end}}{2} \right) + \frac{y_{k,near} + y_{k,end}}{2} \\
 x_k^{abs} &= \frac{m_a m_b (y_{k,ini} - y_{k,end}) + m_b (x_{k,ini} + x_{k,near}) - m_a (x_{k,near} + x_{k,end})}{2(m_b - m_a)} \\
 y_k^{abs} &= y'_a \Big|_{x_k^{abs}} \\
 R_{cir} &= \sqrt{(x_{k,near} - x_k^{abs})^2 + (y_{k,near} - y_k^{abs})^2} \\
 R_{obs} &= \sqrt{(x_{k,near} - x_k^{abs})^2 + (y_{k,near} - y_k^{abs})^2} + R_{robot}
 \end{aligned}$$

수식 1. 장애물의 원형화 과정에 필요한 변수

의 편의와 알고리즘의 간단한 적용을 위해 로봇을 점으로 간주하고 이를 보상하기 위해 장애물을 확대시키는 것이다. 장애물의 원형화 과정을 그림 2에 표기한 변수들을 사용하여 수식 1로 요약하였다.

### 2.1.2 VTV 알고리즘

VTV의 가장 핵심이 되는 개념은 경로계획에 있어서 주행로봇의 이동방향(Heading Direction)을 결정하기 위해 목표점으로부터의 인력 벡터와 원형화된 장애물로부터 얻어진 접선 벡터의 합력을 구하는 것이다. 접선 벡터는 LMS로부터 얻은 원형화된 장애물과 로봇과의 거리가 가장 가까운 점으로부터의 계산이 된다. 또한 인력 벡터의 방향은 현 위치의 주행로봇에서부터 목표점까지이고 크기는 일정한 값을 가진 벡터이다. 주행로봇의 이동방향은 다음과 같은 식으로 표현된다.

$$\vec{D}_{res} = \pm \vec{m}_{k,dir} + \vec{F}_{gravity} \quad (2)$$

여기서  $\vec{D}_{res}$  는 로봇의 이동방향을 나타내고  $\vec{m}_{k,dir}$  는 접선 벡터,  $\vec{F}_{gravity}$  는 목표점으로부터의 인력 벡터를 의미한다. 접선 벡터의 방향은 원형화된 장애물의 로봇과 가장 가까운 점에서의 접선 방향이고, 크기는 아래의 식 (3)과 같이 원형화된 장애물의 반지름에 비례하게 선정된다. 이는 비례상수를 조절함으로써 장애물을 유연하게 피하게 하기 위함이다.

$$|\vec{m}_{k,dir}| = \eta \times R_{obs} \quad (3)$$

지역 국소점 현상을 부분적으로 해결하기 위해 VTV를 두 개의 모드(Mode)와 각 모드에 대해 두 가지 상황(Case)으로 나누었다. 모드 1은 LMS로부터 얻은 세 점 중 최소한 한 점은 일직선상에 존재하지 않는 경우로 정의하고, 모드 2는 세 점을 지나는 선이 일직선인 경우로 정의한다. 상황을 두 가지로 나눈 이유는 목표점까지의 짧은 경로를 생성하기 위한 것으로, n-1 번째의 이동방향 벡터와 현재(n 번째) 접선 벡터와의 내적이 양의 값을 가지면 n 번째 접선 벡터를 그대로 사용하여 새로운 이동방향 벡터를 구하고, 음의 값을 가지면 n 번째 접선 벡터에 음의 값을 취한 반대 방향으로 새로운 이동방향 벡터를 업데이트한다. 이를 수식으로 나타내면 다음과 같다.

$$\begin{aligned} \vec{m}_{k,dir(n)} \cdot \vec{D}_{res(n-1)} > 0 \\ \rightarrow \vec{D}_{res(n)} = +\vec{m}_{k,dir(n)} + \vec{F}_{gravity} \end{aligned} \quad (4)$$

$$\begin{aligned} \vec{m}_{k,dir(n)} \cdot \vec{D}_{res(n-1)} < 0 \\ \rightarrow \vec{D}_{res(n)} = -\vec{m}_{k,dir(n)} + \vec{F}_{gravity} \end{aligned} \quad (5)$$

만약 LMS로부터 얻은 세 점이 일직선상에 존재할 경우(모드 2)에는 원형화된 가상 장애물의 반지름이 무한대의 값을 가지므로 원형화된 장애물의 반지름에 비례하는 접선 벡터의 크기는 마찬가지로 무한대의 값을 가지게 되고, 따라서 항상 일정한 값을 유지하는 목표점으로부터의 인력 벡터는 무시될 수 있다. 이 경우 장애물은 무한히 긴 벽으로 간주되며 주행로봇은 유한한 크기의 접선 벡터가 얻어질 때까지 벽을 따라 움직인다.

$$|\vec{m}_{k,dir}| = \eta \times R_{obs} \approx \infty \quad (6)$$

$$\vec{D}_{res} = \pm \vec{m}_{k,dir} + \vec{F}_{gravity} \approx \pm \vec{m}_{k,dir} \quad (7)$$

VTV의 전체 흐름을 보면 그림 3과 같다.

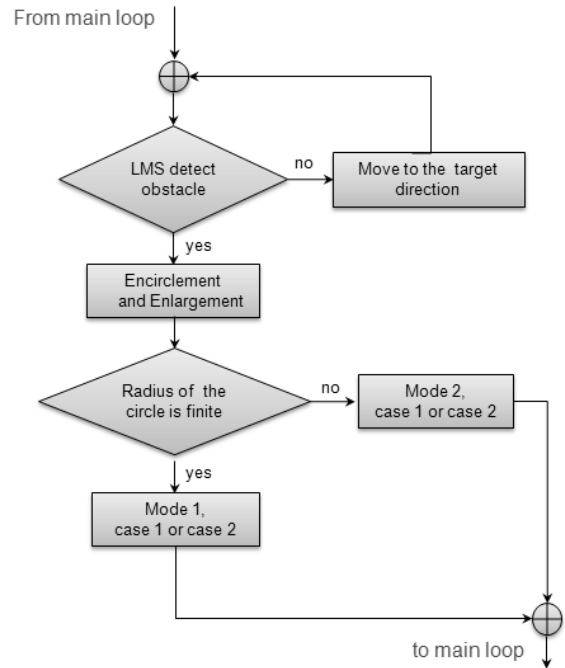


그림 3. VTV 순서도

### 2.1.3 VTV 모의실험

제안된 VTV 방법은 접선벡터를 이용하여 로봇의 방향을 결정하는 방법으로 지역 국소점 현상을 부분적으로 해결하였다는 장점이 있다. 그림 4에서는 U 형태의 장애물에 대하여 로봇이 국소점에 빠지지 않고 빠져나오는 모습을 비슷한 두 환경에서 확인할 수 있다.

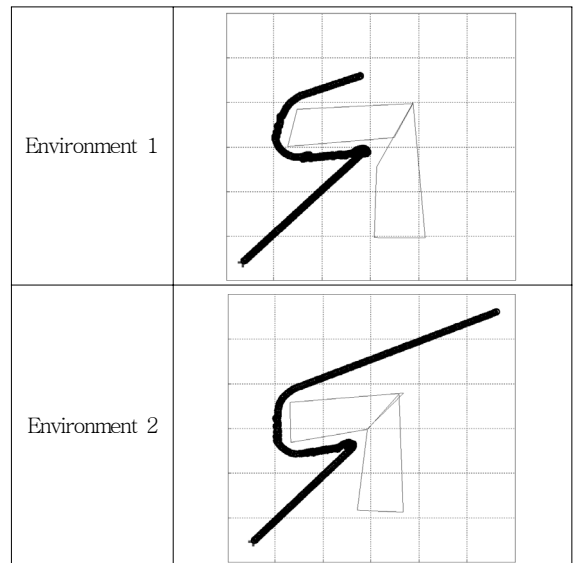


그림 4. U 형태의 장애물 회피

### 2.2 로봇의 안전 속도를 위한 퍼지 제어

일반적인 2D 환경에서 주행로봇이 자율적으로 장애물을 피해 목표점에 도달하는 과정은 장애물 회피 알고리즘 만으로도 가능하지만 예기치 못한 장애물이나 이동 장애물을 만났을 경우에는 추가적인 방법이 필요하다. 특히 이러한 경우에는 안전성 확보를 위해 장애물 회피와 더불어 로봇과 장애물 간 충돌 방지에 무게를 둘 수 있는 실시간 속도 제어기법이 고안되어야 한다. 이에 본 연구에서는 소속 함수(Membership Function)와 규칙 집합(Rule Set) 기반의 퍼지 제어를 통해 주행로봇의 속도를 가변시키는 방법을 제안한다.

#### 2.2.1 퍼지 제어를 위한 소속 함수 정의

속도를 제어하기 위해 제안된 퍼지 제어기에는 로봇과 장애물간 거리, 지면 경사, 지면의 거칠기 등 총 3개의 입력이 있고 각각에 대하여 그림 5와 같이 5개의 소속 함수를 정의하였다. 출력에 해당하는 속력은 7개의 소속 함수로 정의하였고 이는 그림 6과 같다. 입, 출력의 최대, 최소값은 2D 환경에서 VTV를 적용한 결과를 이용하여 경험적으로 정하였다.

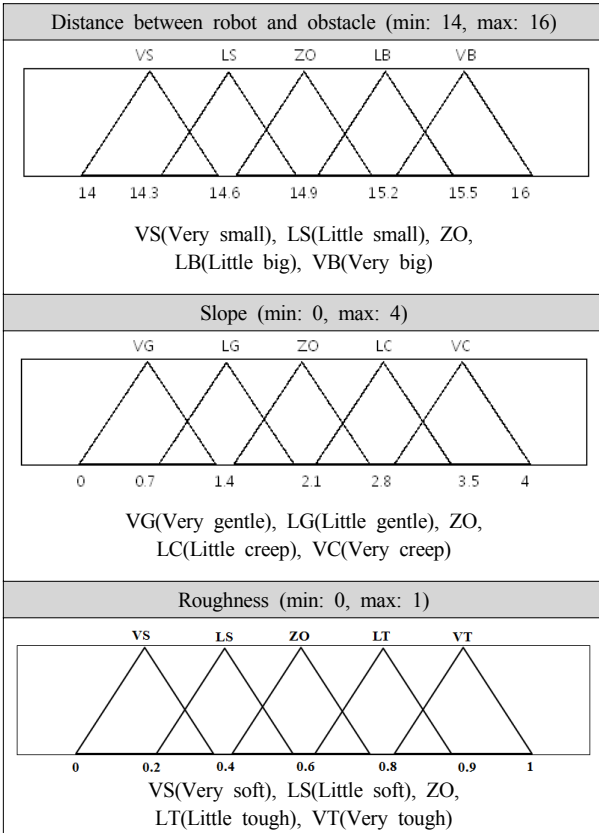


그림 5. 퍼지 입력

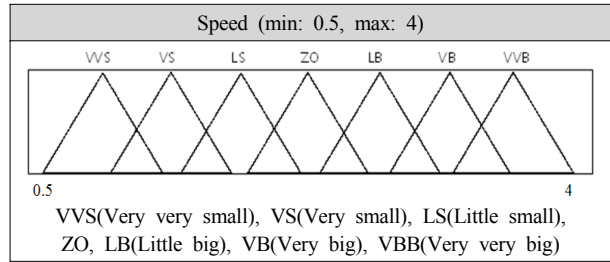


그림 6. 퍼지 출력(속력)

#### 2.2.2 퍼지 제어 규칙 집합

퍼지 제어기 설계 시 규칙 집합을 정하는 과정은 매우 중요하다. 따라서 퍼지 규칙 집합은 숙련된 경험자나 공학자의 경험으로 정하는 것이 바람직하다. 2개 이상의 입력이 존재할 때는 규칙을 정하기가 어렵기 때문에 기준(Standard)을 정하는 것이 좋다. 본 논문에서는 퍼지 규칙을 정하기 위해

- (1) 경사가 급할수록 안전하게 주행
- (2) 지면의 거칠기가 크면 안전하게 주행
- (3) 장애물로부터의 거리가 가까우면 안전하게 주행

표 2. Input: distance(A),roughness(B),slope(C), output: speed

Selected 22 rules	Output (speed)
If input A is VS and input B is VT and input C is VC	VVS
If input A is VS and input B is LT and input C is VC	VVS
If input A is VS and input B is VT and input C is LC	VS
If input A is VS and input B is ZO and input C is ZO	VS
If input A is LB and input B is VS and input C is VG	VVB
If input A is LS and input B is VT and input C is VC	VS
If input A is LS and input B is ZO and input C is LC	VS
If input A is LS and input B is VS and input C is ZO	LS
If input A is LB and input B is VS and input C is VG	VB
If input A is ZO and input B is LF and input C is ZO	ZO
If input A is VS and input B is ZO and input C is VC	VS
If input A is LB and input B is VS and input C is VG	VB
If input A is VS and input B is ZO and input C is VC	VS
If input A is LS and input B is LT and input C is VC	VVS
If input A is ZO and input B is VS and input C is ZO	LB
If input A is ZO and input B is VS and input C is LG	ZO
If input A is ZO and input B is LT and input C is VC	VS
If input A is VB and input B is VS and input C is VG	VVB
If input A is VB and input B is LF and input C is ZO	LB
If input A is VB and input B is ZO and input C is ZO	LB
If input A is LS and input B is LT and input C is VC	VVS
If input A is LB and input B is VS and input C is VG	VB

하도록 하는 기준을 정하였다. 또한 세 가지 입력을 충돌에 대한 안전성과 접목하여 다음과 같이 우선순위를 선정하였다.

$$\text{Slope(경사)} > \text{Roughness(거칠기)} > \text{Distance(거리)}$$

기본적으로 로봇과 장애물간 거리가 가장 위험성이 크지만 제한된 VTV에서 장애물을 회피하여 경로계획을 하기에 마지막 우선순위로 정하였다. 정한 기준으로 퍼지 규칙을 표 2와 같이 정의하였다. 입력이 3개이고 각각에 대한 소속 함수가 5개이기 때문에 총 125개( $5^3$ )의 규칙이 있어야 하지만 중복되거나 불필요한 규칙을 제외하고 모의실험에 크게 영향을 미치는 요소 22개를 최종 퍼지 규칙 집합으로 상정하였다.

### 3. 통합 시스템

통합 시스템은 크게 GPS, IMU와 LMS를 통하여 로봇의 위치와 자세, 장애물로부터의 거리를 얻어 연산을 수행하는 VTV와 주어진 데이터(경사, 거칠기)를 이용하여 속력을 제어하는 퍼지 제어기로 그림 7과 같이 구성된다.

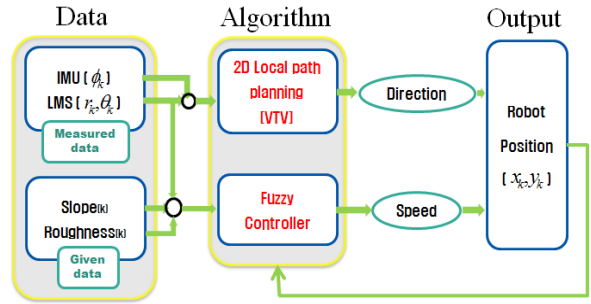


그림 7. 통합 시스템

### 4. 모의실험 결과

제한된 알고리즘을 검증하기 위하여 Matlab을 이용하였으며 경사 데이터를 부여하기 위하여 Matlab에 내장된 함수인 Peak Function을 사용하였다. 모의실험을 통하여 VTV와 로봇의 속력이 퍼지 제어를 통해 적절하게 출력되는지를 확인하였다. 로봇의 크기는 2m(W) × 3m(L) 이고 센서는 4m, 270° 스캔 가능한 사양으로 가정하여 모의실험이 수행되었다.

그림 8의 Amplitude Information은 3D로 본 지도이고 밑의 두 그래프는 로봇이 주행한 경사도와 속도 값이다.

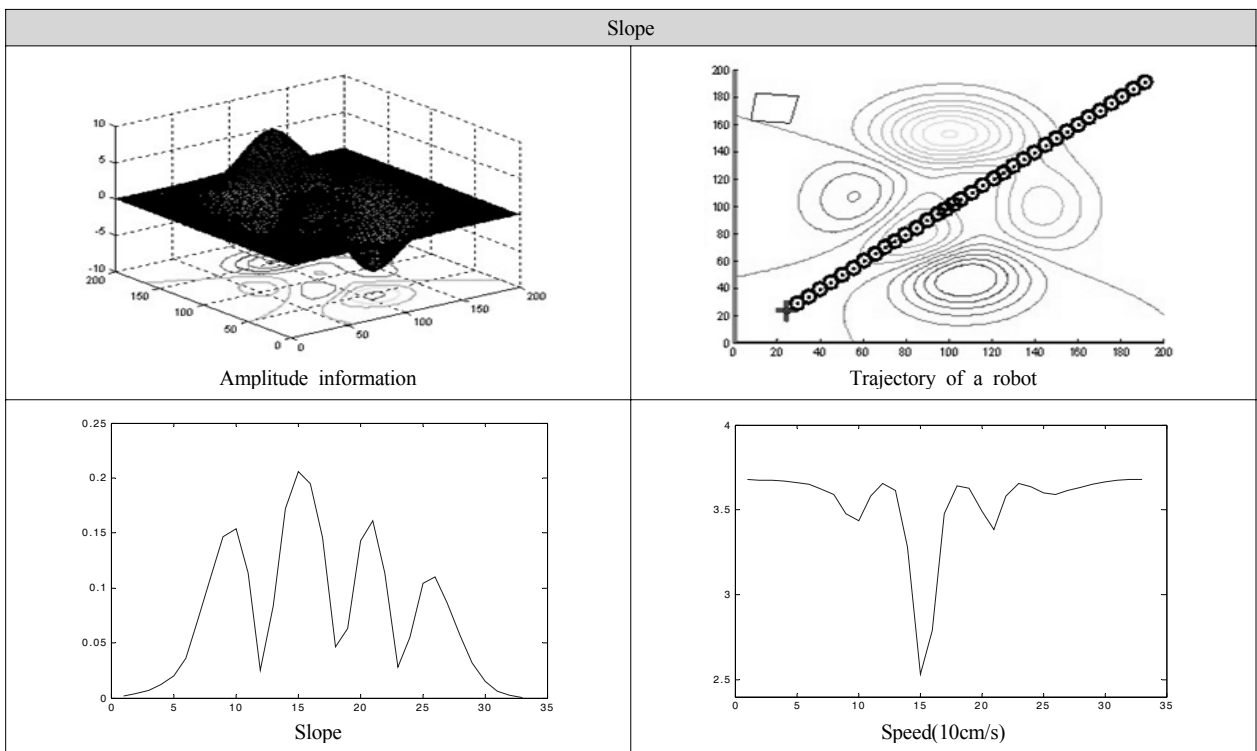








그림 8. 기울기 영향에 의한 속도 변화

표 3. Roughness notations

	0.8		0.5		0.4
	0.1		0.2		0.6

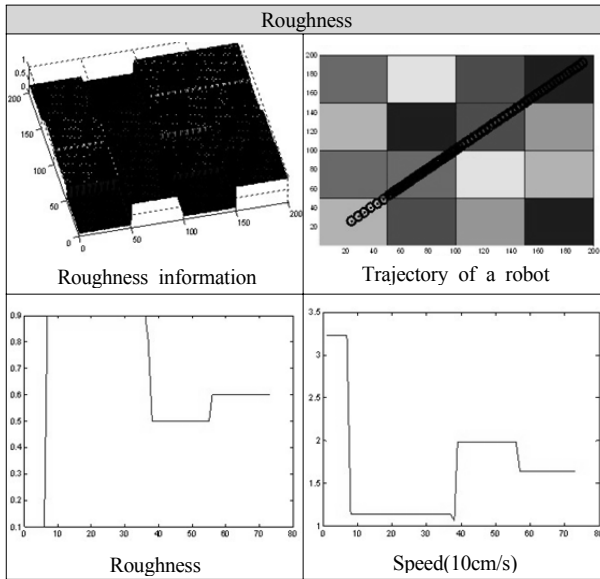


그림 9. 거칠기 영향에 의한 속도 변화

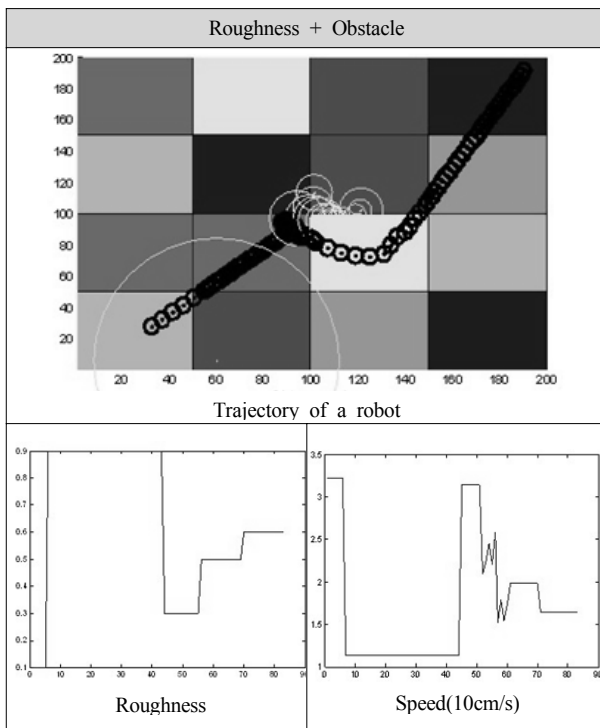


그림 10. 거칠기와 장애물 영향에 의한 속도변화

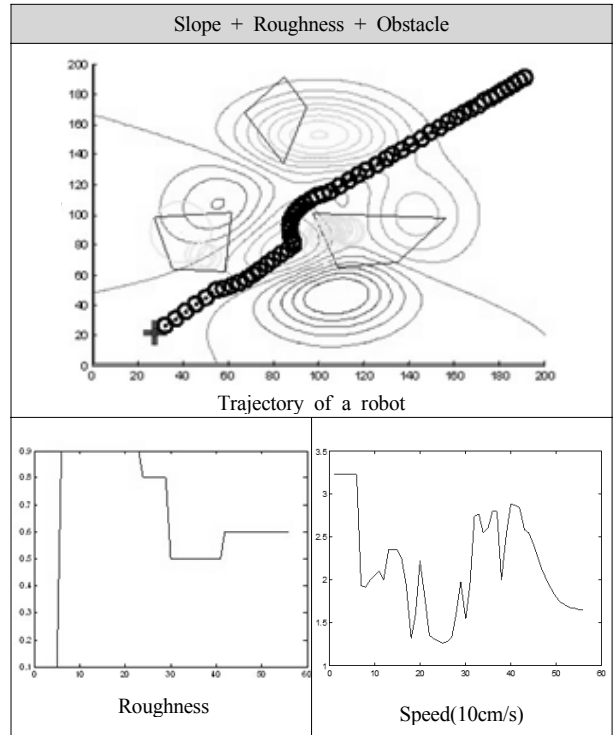


그림 11. 기울기, 거칠기, 장애물 영향에 의한 속도 변화

표 3은 퍼지 제어가 지면의 거칠기에 잘 적용이 되는지 검증하기 위해 정의된 거칠기 코드이다. 색깔에 따라 거칠기를 0부터 1사이의 값으로 부여하였다. 그림 9는 지면의 거칠기에 따른 속력의 변화를 확인한 것으로 거칠기와 속력의 경향이 서로 반대임을 관찰함으로써 퍼지 제어가 잘 작동되는 것을 확인할 수 있었다. 마찬가지로 그림 10은 로봇과 장애물로부터의 거리와 지면의 거칠기에 대한 속도 변화가 퍼지 제어를 통해 적절히 출력되는지를 검증한 결과이다.

이때 속도 변화 그래프에서 속력이 떨어지는 부분이 있는데 이는 장애물의 영향으로 안전하게 로봇이 주행하기 위함이다. 그림 11은 지면의 경사, 거칠기와 장애물이 모두 고려된 경우이며 역시 속도 값이 퍼지 제어를 통해 적절히 제어되는 것을 확인할 수 있었다.

### 5. 결론

본 연구에서는 주행로봇이 목표점을 향해 경로계획을 수행할 때 필요한 장애물 회피 알고리즘 VTV를 제안하고 로봇의 안전을 위한 퍼지 제어를 추가하여 로봇의 속력을 제어하였다. 제어 대상이 로봇의 구동이 아닌 지면이나 장애물 등의 환경을 대상으로 제어하도록 하는 새로운 방법을 제안하였다. 제안된 준 3차원 경로계획 알고

리즘은 모의실험을 통하여 VTV와 퍼지 제어가 적절히 운용되는지 검증하였다. 모의실험 결과, 입력 요소에 대해 우선순위가 부여된 퍼지 제어를 통해 로봇의 속력이 가변되는 것을 확인하였고, 이로써 다양한 환경 변수에 대해서도 주행로봇의 안전이 확보 가능함을 알 수 있었다. 그러나 퍼지 입력과 퍼지 출력의 소속 함수 범위를 변화시킴에 따라 출력 속력 및 장애물 회피 시 장애물과의 거리가 조금씩 달라지는 현상이 발생하였다. 따라서 향후에는 이러한 현상을 해결하고 최적의 파라미터 값을 얻는 방법에 대한 연구가 진행되어야 할 것이고, 본 연구에서 개발/검증된 경로계획 알고리즘을 이용하여 주행로봇의 국소 지역 자율주행을 실험적으로 검증하는 연구가 수반되어야 할 것이다.

Conference on Control, Automation and Systems, Seoul, Korea, pp.1597-1600, 2008.



**곽 경 운**

2005 중앙대학교 기계공학과 (공학사)  
 2006~현재 KAIST 기계항공시스템학부 기계공학전공 박사과정  
 관심분야: 주행로봇의 경로계획 및 장애물 회피

**참고문헌**

[1] J. Borenstein, "Real-time obstacle avoidance for fast mobile robots," IEEE Transactions on Systems, Man, and Cybernetics, Vol.19, No.5, pp.1179-1187, Sept./Oct., 1989.

[2] J. Borenstein, Y. Koren, "the vector field histogram-fast obstacle avoidance for mobile robots," IEEE Journal of Robotics and Automation Vol.7, No.3, pp.278-288, June, 1991.

[3] Dieter Foxy, Wolfram Bardy, Sebastian Thrunyz, "The Dynamic Window Approach to Collision Avoidance," IAI-TR-95-13, August, 1995.

[4] J. Minguez, "Nearness Diagram (ND) Navigation: Collision Avoidance in Troublesome Scenarios," IEEE Transactions on Robotics and Automation, Vol.20, No.1, pp.45-59, 2004.

[5] 정해관, 현경학, 김수현, 곽윤근, "구조로봇에 적합한 장애물 회피 알고리즘 ELA의 실험적 검증," 한국로봇공학학회논문지, 제 4권, 제 2호, pp.105-111, 2009.

[6] 변증남, "퍼지논리 제어(Fuzzy-Logic Control)", 홍릉과학 출판사, 1997.

[7] Driankov, Hellendoorn, Reinfrank, "An Intro. to Fuzzy Control," Springer-Verlag, 1993.

[8] Kyung Woon Kwak, Hae Kwan Jeong, Soo Hyun Kim and Yoon Keun Kwak, "VTV: Real Time Obstacle Avoidance of Mobile Robots for Local Path Planning using LMS," International



**정 해 관**

2003 KAIST 기계공학과(공학사)  
 2005 KAIST 기계공학과(공학석사)  
 2009 KAIST 기계항공시스템학부(공학박사)  
 2010~현재 연구개발인력교육원 기획연수 1팀  
 관심분야: 주행로봇의 장애물 극복 및 회피



**김 수 현**

1978 서울대학교 기계공학과 (공학사)  
 1980 KAIST 기계공학과(공학석사)  
 1991 Imperial College, Univ. of London(공학박사)  
 1991~현재 KAIST 기계항공시스템학부 교수  
 관심분야: 로봇 메커니즘 설계 및 제어