論文 89-26-7-5

# Structured Light을 이용한 이동 로보트의 3차원 환경인식

# (Recognition of 3-Dimensional Environment for a Mobile Robot Using Structured Light)

李 碩 埈*, 鄭 明 振**

(Seok Jun Lee and Myung Jin Chung)

**要 約**

본 논문은 이동 로보트가 실제 환경속에서 주행할 수 있도록 하기 위한 간단한 structured light 센서의 개발과 이용에 관한 연구로써, 이동로보트에 이 센서를 부착하였을 경우 그 적용범위는 첫째, 복잡한 3차원 물체들속에서의 실시간 주행과 둘째, 주행경로 주변 물체의 모델링과 인식을 위한 것으로 나뉠 수 있다. 개발된 센서가 갖추고 있는 데이타 습득 속도와 정확도는 실용화 하기에 충분하며, 여러 다양한 상황속에서의 실험을 통하여 이를 확인하였다.

**Abstract**

**In this paper, a robust and simple structured light sensory system has been studied to endow mobile robots with the ability of navigating in real world. A mobile robot with this sensor can be applied in two ways: first, real time navigation in 3-dimensional world, second, modeling and recognition of environment. Range data obtained with this sensor are fairly accurate, and the data aquisition speed is satisfactory. Experiments in diverse situation show effectiveness of the structured light sensor for the mobile robot.**

## I. 서 론

과거 단순 조립과 가공을 위해 산업현장에 처음 도입되기 시작한 로보트는 오늘날 전자공학과 기계공업의 발전속에서 그 제작, 운용 기술이 급속히 발달하게 되었고, 더불어 고도 산업사회에서 생산력 증가와 인건비 절감을 추구하는 기업들의 수요도 급증하고 있어, 여러 다양한 분야에서 그 생산적 비중을 점차 키워 나가고 있다.

이러한 흐름속에서 로보트의 사용 용도와 작업환경이 다양해져 가고 있고, 그에 따른 로보트의 적응능력에 대한 사용자측의 요구도 여러가지로 나타나고 있어서 요즈음은 로보트에 여러 종류의 센서를 부착해 제어함으로써 자체 인식과 판단기능을 갖는 지능로보트를 구현하려는 노력들이 부단히 이루어 지고 있다.

이동로보트(mobile robot)의 출현도 이와같은 맥

*準會員, 金星産電(株) 研究所
(GoldStar Industrial Systems Co., Ltd.)
**正會員, 韓國科學技術院 電氣 및 電子工學科
(Dept. of Electrical Eng., KAIST)
接受日字 : 1989年 4月 8日

락에서 이루어 졌다고 하겠다. 즉, 로보트가 주어진 환경속에서 작업을 할 경우 그 작업 공간(work space)은 자연히 제한받기 마련인 바 로보트의 몸체에 이동 능력을 부여하여 작업 공간을 임의로 변경 확장할 수 있도록 하자는 것이 그 출발점이었다. 그리고, 지정된 경로를 따라가는 단순한 이동 기능에서 한발 더 나아가 심해나 원자력 발절소등과 같이 사람이 접근하기에 위험한 장소를 탐사하고, 그 속에서 작업할 수 있는 자동주행로보트를 개발하려는 연구들이 속속 수행되고 있다.

실제로 로보트가 특정 경로나 목표점으로 이동할 때든지, 도중의 방해물과의 충돌을 피하기 위해 경로를 변경해야 할 경우, 그리고 궁극적인 작업 대상물과 환경의 인식을 위해선 우선 주변에 존재하는 물체까지의 거리를 측정할 수 있는 센서가 필요하다.

지금까지 로보트를 대상으로 개발된 많은 센서들 중에 어떤것을 선택할 것인가는 이동로보트의 성능과 작업조건, 사용자가 원하는 정확도 그리고 거기에 따르는 비용이 함께 고려되어져야 할 것이다.

본 논문에서는 연구목표를 우리 주변의 실제 3차원적 환경속에서 로보트가 방해물과의 충돌없이 주행할 수 있도록 함과, 이를 복잡하고 값 비싼 장비없이도 실시간에 가깝게 구현하는 것으로 했다.

따라서 여기에는 충분한 정확성을 갖는 3차원 거리값들이 되도록 빠른 시간에 얻어져야 된다는 것과 실제로 실용화될 수 있도록 센서 설치에 드는 경제적 부담을 최소화시켜야 한다는 필요가 생긴다. 이와 같이 서로 상충되는 필요조건을 최대한 만족시키는 방법으로 structured light 센서체계를 선택하였다.

원래 structured light 기법은 computer vision이나 machine intellegence 분야에서 제한된 환경속의 간단한 물체인식을 위해 처음 시작하였고[1,3,4], 그 뒤 좀더 효과적인 데이타 처리와 다양한 물체의 인식을 겨냥한 연구들이 뒤따라 발표되었다. 이 structured light 방식의 거리 센서가 실용화를 겨냥해 로보트 분야에 도입된 것은 최근의 일인데, 그 동기는 2가지로 나뉠 수 있다. 우선 조립 작업을 수행하는 로보트 머니퓰레이터를 위한 장치로써 개발이 이루어 졌다. 1985년 Agin[5]은 PUMA 머니퓰레이터의 작업을 위한 보조기구로 레이저 다이오우드(laser diode)와 카메라로 구성된 간단한 structured light 센서를 고안하였는데, 이는 일정한 작업 공간내에 소형 물체들의 인식을 위해 structured light을 이용한 시도이다. 두번째로는 이동로보트의 경로 결정을 위해 주변환경에 관한 정보를 얻는데 이용하려는 방향이다. 1987년 Brown[6]은 stripe 형태의 빛을 수평방향으로 연속적으로 주사함으로써 야외의 풀밭에서 이동로보트가 자신의 길을 찾아내는 방식을 제안하였고, 1988년 Moign과 Waxman은 이동로보트의 주행을 위해 grid 형태의 structured light를 주사하고, 이의 인식을 위한 소프트웨어 기법의 개발을 주제로 한 논문을 내놓았다.

본 논문에서 가정하는 상황은 이동로보트가 주변 물체에 대한 정보가 없는 상태로 현재의 위치에서 목표점으로 이동해 나가는 것으로 한다. 이러한 가정하에서 논문의 목표를 다음의 세가지로 요약할 수 있다.

1. 3차원적인 실제 환경에서의 주행

이동로보트에 관한 지금까지의 많은 논문들이 주행 알고리듬의 simulation에 머물거나, 미리 인공적으로 조성된 환경속에서 움직여 나가는 것으로 대신하는 상태이다. 그러나 이동로보트를 위해 structrued light 센서를 부착한다면 실제 생활 주변에 존재하는 복잡한 3차원적 물체속에서도 로보트가 충돌없이 주행할 수 있다.

2. Personal computer을 이용한 실시간 주행

로보트의 이동을 위해 값 비싼 센서를 필요로 하거나, 데이타의 실시간 처리를 위해 과도한 용량의 컴퓨터가 있어야 한다면 그 이동로보트의 실용화에는 분명한 한계가 있을 수 밖에 없다. Structured light 센서에 의해 얻어지는 데이타를 몇가지의 사전 조작을 거쳐서 현재의 IBM-AT 정도의 컴퓨터를 이용해 실시간에 가까운 경로 결정을 해내고자 한다.

3. 주변 환경의 기하학적 정보 추출과 인식

이동로보트가 미지의 환경속을 움직여 나간다면 단순히 충돌을 피할 수 있도록 자신의 경로를 찾아가는 것 뿐아니라 그 환경을 인식, 모델링하는 기능도 지니고 있어야 한다. 본 논문은 매 순간 이동로보트에 들어오는 주변물체에 관한 데이타에서 그 기하학적 정보를 효과적으로 추출, 저장하는 방법에 관해서도 연구를 진행시켰다.

## Ⅱ. 전체 시스템의 구성

1. 하드웨어의 구성

본 연구에서 구성된 이동로보트를 위한 structured light 센서 체계는 그림 1과 같다.

1) Stripe 주사부

Stripe 주사부는 7mW HeNe laser와 렌즈, 거울, 지지장치 그리고 모터와 드라이브 회로로 구성되어 있다. Structured light 주사방식은 띠(stripe) 형태로서 이동로보트의 상단부에 위치하는데, 로보

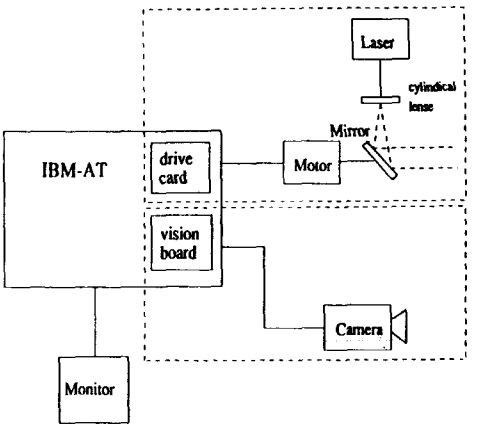



그림 1. 이동로보트를 위한 structured light의 센서의 구성

Fig. 1. The configuration of structured light sensor.

트의 주행경로를 실질적으로 제한하는 것은 로보트의 최고점에서 지면사이에 있는 물체이므로 그 경계면을 쉽게 결정해 주기 위함이다.

Cylindrical 랜즈를 통과한후 형성된 띠 형태의 빛은 최종 반사 거울에 반사되면서 지면과 일정 각도로 교차되는 수평방향의 평면이 된다. Stripe을 수평방향으로 비추는 이유는 일반적 상황에서 물체들의 수평방향 변위가 수직방향보다 변화가 심할 뿐더러, 우선적으로 로보트의 이동을 제한하는 것은 평면적인 물체 분포이기 때문이다. 일단 매 순간 stripe이 비추는 곳의 데이타가 저장되면, 사용자가 미리 프로그램한 방향과 간격으로 최종 반사거울을 회전하여 stripe를 수직으로 이동시키고 이러한 반복 작업을 통해 앞쪽 전면에 관한 정보를 축적해 간다. 물론 이 방식은 grid 형태로 쓰는 것에 비하면 상대적으로 데이타 습득 속도가 느리지만 주행 환경에 따라서 빛을 비추는 간격을 적절하게 조정해 줄 수 있고 광학적 관점에서 센서 구성이 간편하다는 장점이 있다.

2) Stripe pattern 인식부

이 부분은 카메라와 vision board로 구성되어 있다. 대상 물체들에 맺힌 stripe 형태를 감지하고 이를 thresholding과 디지탈화하는 작업을 수행한다.

실제 시스템에서 사용된 것은 가시광선 영역의 C-CD 카메라와 Frame Grabber vision board인데 이들은 모두 일반적 용도로서 생산된 완제품이라서 structured light 센싱방식에 적당한 것들로 교체된다면 시스템 전체의 수행 효율을 훨씬 증대시킬 수 있을 것이다.

이와 같은 stripe 주사부와 stripe 형태 감지 이외의 모든 작업은 IBM-PC/AT에서 소프트웨어적으로 이루어 진다.

2. 소프트웨어 개요

본 연구에서 이동로보트의 이동과 물체 인식을 위하여 개발된 소프트웨어는 다음과 같이 분류될 수 있다.

1) 카메라 calibration과 3차원 데이타 습득

이 과정은 3차원 공간과 카메라 image 평면과의 대응관계를 결정하도록 카메라의 변환 행렬을 구하며 이어서 laser stripe의 평면 방정식과 연립하여 구체적인 거리 데이타를 얻는다.

2) 금지영역 확인과 경로결정

이동로보트의 실질적 주행을 가능하게 해주는 부분이다. 전방에 존재하는 물체에 대한 거리를 구하고 이 물체의 3차원적 형태를 지면으로 투영시켜 금지영역을 설정한다. 그리고 로보트가 지나갈 수 있는 공간을 찾아 경로를 결정하며 동시에 여기서 얻어진 정보로서 로보트의 금지영역을 알려주는 평면도를 저장한다.

사용자가 주행속도에 특히 관심이 있다면, 아래의 모델링 과정을 생략하고 1)과 2)만으로 로보트의 주행이 가능하다.

3) 주변환경에 대한 모델링

이 부분에서는 로보트 주변에 대해 얻어진 데이타를 효과적으로 축약, 일정한 구조로 저장하며 이 초기 데이타에서 평면과 관계된 정보를 얻을 수 있다면 그에 수반된 후속 작업도 진행한다. 본 논문에서는 기본적으로 어떠한 평면이 존재하는가를 찾아보고 존재한다면 초기 데이타를 평면을 규정하는 다른 형태의 데이타 구조로 변환시킨다.

## Ⅲ. 3차원 거리 데이타 습득

1. 카메라 calibration

카메라를 calibration한다는 것은 2차원인 카메라 image 평면으로 주변의 3차원적인 공간이 mapping되는 관계를 알아내는 것이다.

그림 2와 같이 공간의 일정 지점에 기준 좌표축이 설정되고 이 좌표축의 특정 지점과 방향으로 카메라가 놓여 있다면 공간내의 임의의 한 점과 거기에 대응되는 image 상의 한 점은 아래와 같은 벡터 형태로 표현된다.

$$W=[X\ Y\ Z]^T \quad C=[x\ y]^T \qquad (1)$$

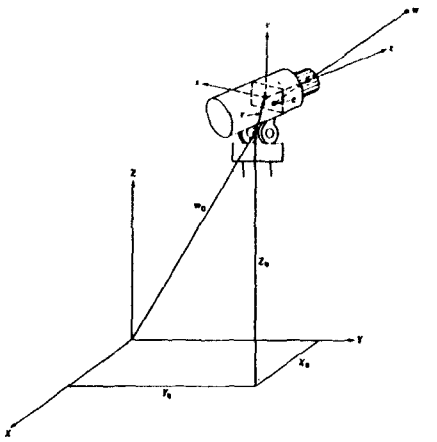


**그림 2.** 기준 좌표축과 카메라 좌표축과의 관계
**Fig. 2.** Imaging geometry with two coordinate systems.

이들을 homogeneous coordinates 형태로 나타내면

$$W_h=[kX\ kY\ kZ\ k]^T\quad C_h=[kx\ ky\ kz\ k]^T \qquad (2)$$

$W_h$와 $C_h$의 mapping 관계, 즉 두점간의 변환 행렬(transformation matrix)은 카메라 초점거리, offset, pan각과 tilt각을 측정해 줄 수 있지만, 이와 같은 파라미터들을 측정하는 것이 쉽지 않기 때문에 이미 공간상의 좌표값을 아는 몇개의 점과 그 점들의 image 평면상의 대응 좌표값을 가지고 역으로 계산하는 방법이 많이 쓰인다.

k=1로 하면 $C_h$와 $W_h$의 관계는

$$\begin{bmatrix}C_{h1}\\C_{h2}\\C_{h3}\\C_{h4}\end{bmatrix}=\begin{bmatrix}a_{11}&a_{12}&a_{13}&a_{14}\\a_{21}&a_{22}&a_{23}&a_{24}\\a_{31}&a_{32}&a_{33}&a_{34}\\a_{41}&a_{42}&a_{43}&a_{44}\end{bmatrix}\cdot\begin{bmatrix}X\\Y\\Z\\1\end{bmatrix} \qquad (3)$$

image 평면에서의 좌표값은 아래와 같다.

$$x=\frac{C_{h1}}{C_{h4}},\quad y=\frac{C_{h2}}{C_{h4}} \qquad (4)$$

따라서 $C_{h1}=xC_{h4}$, $C_{h2}=yC_{h4}$를 (3)에 대입하면, 최종식은 12개의 변수로 이루어진 2개의 식으로 표현된다.

$$a_{11}X+a_{12}Y+a_{13}Z-a_{41}xX-a_{42}xY-a_{43}xZ-a_{44}x+a_{14}=0 \qquad (5)$$

$$a_{21}X+a_{22}Y+a_{23}Z-a_{41}yX-a_{42}yY-a_{43}yZ-a_{44}y+a_{24}=0 \qquad (6)$$

결국 image 평면의 6개 이상의 점에 대해 실제 3차원적 공간 좌표값을 안다면 변환 행렬을 구할 수 있다. 실제로 카메라가 이상적인 pin-hole 형태로 모델링되지 않은 오차요인을 반영하기 위해 보통 20개 이상의 점을 준다.

논문에서의 구체적인 calibration 작업은 다음과 같다.

우선 로보트의 특정 지점을 원점으로 하는 로보트 좌표계를 설정한다. 이동로보트 경우 거리 데이타값은 범위가 수십 cm에서 수 m까지 매우 넓으므로 가장 정확한 값을 얻고자 하는 위치에 기준 물체(sample object)를 놓는다.(그림 3)

**그림 3.** 기준 물체
**Fig. 3.** Sample objects.

그리고 이 물체 표면에 있는 기준점(검은 반점)들의 로보트 좌표계에 대한 위치값을 입력하고, 카메라에 잡힌 image에서 검은 반점의 중심좌표를 구해 calibration한다. 이 중 많은 부분이 소프트웨어적으로 처리된다.

2. 거리 데이타 습득

Structured light 센서를 통해 3차원 데이타를 얻는 과정은 우선 레이저 stripe을 일정 각도 $\theta$만큼 회전시킨 다음, 물체에 맺히는 stripe 형태를 카메라로 잡는데서 시작된다.

그리고 이 stripe pattern을 thresholding과 thinning을 거친 후 stripe이 맺힌 각 화점의 x, y값을 찾는다. 실제 이 과정은 물리적으로 연속된 stripe을 카메라의 수평방향의 화점 수 만큼 sampling하는 의미를 띤다. 실험에 쓰인 카메라의 수평방향의 화점 수

는 512개이므로 카메라의 수평가시각(horizontal view angle)속에 들어오는 stripe을 512개로 나누어 읽는 것이다.

일정한도 $\theta_i$에서 stripe이 맺힌 화점 $P(x_p, y_p)$는 앞에서 구한 변환 행렬 A에 의해 다음의 2개 식을 유도한다.

$$(a_{11}-x_pa_{41})X+(a_{12}-x_pa_{42})Y+(a_{13}-x_pa_{43})Z+(a_{14}-x_pa_{44})=0 \quad (7)$$

$$(a_{21}-y_pa_{41})X+(a_{22}-y_pa_{42})Y+(a_{23}-y_pa_{43})Z+(a_{24}-y_pa_{44})=0 \quad (8)$$

stripe의 각도를 조절하는 모터 축의 위치가 $(X_m, Y_m, Z_m)$이라면 stripe 평면의 식은

$$Z=Z_m-(Y-Y_m)\tan\theta_i \quad (9)$$

따라서 위의 (3)식에서 카메라 image 평면 $(x_p, y_p)$에 관측되는 지점의 공간상의 실제 위치값 $(X, Y, Z)$가 구해진다.

### 3. 측정 정확도 분석

거리 측정값은 로보트에서 2.5m 떨어진 점을 기준으로 했을 때 최소 0.5cm에서 3cm까지의 오차가 있었다.(특히 Y축, 즉 깊이(depth) 방향의 오차가 크다.) 이는 다음과 같은 몇가지 이유로 설명할 수 있다.

첫째, 카메라 렌즈가 이상적일 수 없다는 것이다. 쉽게 가정할 수 있는 것처럼 pin-hole 형태가 아니라 렌즈를 통해 빛이 집속되므로 공간과 image 평면과의 mapping 관계에 왜곡이 일어나며, 이는 특히 image 평면 중앙에서 멀리 있는 화점일수록 심하다.

둘째, thinning 과정에서의 오차이다. 레이저 stripe이 어떠한 물체에 맺혔을 때 이것이 아주 집속된 형태가 아닌 이상 카메라의 인접한 여러 화점에 잡히게 되고 thinning 과정에서 이들의 중앙을 선택한다 해도 1화점 정도의 차이는 쉽게 생긴다.

셋째, 지면의 굴곡과 stripe 면의 misalignment이다. 실제 실험이 건물내부의 여러곳에서 이루어 졌는데 이 경우 건물의 바닥을 z=0인 수평면으로 잡았다. 그러나 건물 바닥은 부분적으로 요철이 있기 마련이다. 그리고 stripe 평면이 x방향의 기울기가 없는 상태로 주사된다는 가정은 cylindrical 렌즈를 통과한 빛이 반사경에 입사할 때 그 반사점들이 모터 회전축의 연장선상에 있어야 성립되나 이것이 완전하게 이루어지는 걸 기대하기는 어려웠다.

## Ⅳ. Stripe 데이타 축약

이동로보트의 전면에 존재하는 물체들에 대하여 stripe을 일정 각도씩 회전하면서, 매 순간 거리 데이타를 얻고 이를 저장한다고 가정해 보자. 이 경우 stripe의 주사 각도가 충분히 조밀하다면 전방 물체에 대한 자세한 정보를 저장할 수 있다. 그러나 $\Delta\theta$씩 변화하면서 $\ell$개의 stripe이 주사되고, 카메라의 화점 구성이 m×n이라면 전체 거리 데이타의 수는 $\ell\times n$개가 될 것이다.

이렇게 되면 앞에서도 언급했듯이 데이타 자체의 크기도 무척 클 뿐더러, 이를 처리하고 인식함에 있어 거기에 소요되는 시간과 컴퓨터의 용량이 매우 부담이 된다. 그리고 이동로보트가 많은 지점을 거쳐 주행한다면 이러한 식의 데이타를 무한정 저장할 수도 없는 것이다.

따라서 특정 각도 $\theta_i$의 stripe에서 거리 데이타가 얻어지면 stripe 평면을 회전시키기 전에 이를 적절한 형태로 축약하는 것이 바람직하다. 물론 이로 인해 약간의 정보를 잃는다 하더라도 경로 결정에 걸리는 시간의 경감과 메모리의 절약이 이동로보트의 입장에서는 훨씬 중요하다.

데이타는 다음과 같은 2가지 방식으로 축약된다.

### 1. 기준면으로의 투영(Projetion)

보통 실내나 실외에서 운용되는 이동로보트는 지면이나 건물의 바닥 위(floor)를 이동한다. 따라서 주변 물체들의 3차원적 형태를 이러한 기준면(지면, 건물바닥)으로 투영시키면, 적어도 각 물체의 형태상 이동로보트의 움직임을 제약하던 부분은 투영된 도형에 반영이 된다.

이 방법은 일정 각도의 stripe에서 얻어지는 위치값에서 높이 방향의 좌표값을 무시한 후, 이 거리값들을 range buffer에 저장되어 있는 그 이전 주사각도에서의 stripe 좌표값들과 비교를 한다. 그리고 새로 얻어진 데이타가 대상물체의 영역이 확장되었음을 의미할 때는 range buffer에 저장되는 값을 교체한다. 그리고 $\Delta\theta$만큼 stripe을 회전하여 반복한다.

이와 같이 얻어진 range buffer내의 최종값은 대상물체가 기준면으로 투영되었을 때의 경계를 형성하며, 이 정보를 가지고 다음 이동경로를 선정하면 주변 물체와의 충돌은 없으리란 것을 보장 받을 수 있다. 이 투영방법에 의한 경로 결정은 로보트 진행방향의 환경을 얼마나 조밀하게 stripe으로 조사하느냐에 달린 문제이지만, 필요로 하는 계산양이 적다는 점과, 데이타를 저장하는 방식상 실제 환경에서 실시간 경로 결정을 가능케하는 매우 적합한 방법이다.

### 2. 선 분할(Line Segmenting)

Ⅲ.2절에서 설명했듯이 어떠한 stripe에 의해 비추

어진 각 물체들까지의 거리 측정은 카메라의 수평방향의 화점 수, 즉 512개만큼 sampling되어서 이루어진다. 즉 한번 stripe 주사에 의해 공간의 512개의 점의 좌표값이 얻어지는 것이다.

이와 같이 얻어지는 512개의 점을 3차원 공간상의 몇개의 선으로 묘사하여 그 데이타양을 줄여 저장할 수 있다. 그런데 2차원 평면상의 점들을 몇개의 선으로 근사화하는 방법은 알려져 있다.[6,12]

본 논문에서는 이와 같은 알고리듬을 간단히 3차원으로 확장해 이용하였다.

1) 선에 의한 근사화

실제로 그 위치가 계산된 점의 분포가 그림 4 와 같다면 몇개의 선으로 근사화되는 과정은 아래와 같다. 우선 A점과 B점을 연결하는 선을 가정한다. 그리고 실제로 존재하는 점들과 선분 AB와의 거리를 구하고, 오차라고 인정할 수 있는 이 거리값($w_1$)이 어떠한 한계값(error bound)를 넘어서면 다시 AB에서 가장 멀리 떨어진 점 C와 A, B점을 끝점으로 하는 선분 AC, CB를 가정하고, 선분 AC와 A와 C사이의 점, 선분 CB와 B와 C사이의 점들에 대하여 위 과정을 반복한다. 그리고 또 그 어느 구간이 오차 한계를 넘어서면 그 구역에서 최대 오차를 갖는 점(예를 들어 AC구간의 D점)을 새로운 분기점으로 하는 두 선분(AD와 DC)를 새로 설정한다. 이와 같은 반복 작업을 통해 그림 4 에서 공간상의 23개 점은 최종 6개의 선분으로(AD, DE, FC, CG, HI, IB) 묘사된다.

실제 점들의 위치에 이를 근사화하는 선들이 어느정도 일치하느냐는 사용자가 오차 한계값을 어느정도 작게 잡느냐에 달렸다. 그런데 이 값을 너무 작게주면 계산시간이 길어지고, 데이타양 축약 효과가 별로 없다.

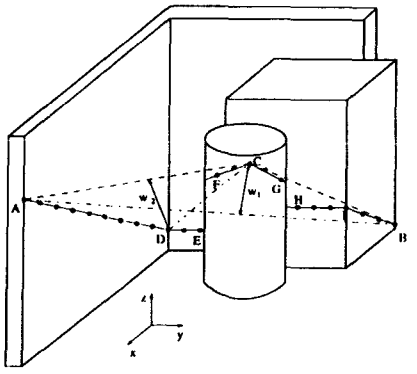



**그림 4.** 선 분할 예제
**Fig. 4.** Example of line segmenting method.

2) Stripe의 3차원적 분할

실험에서 1개의 stripe에서 얻어지는 3차원 위치값은 512개라 했다. 이 경우 대상 물체들간의 불연속성계(jump edge)에 해당되는 점은 선 분할 과정에서 반드시 분절점으로 선택된다고 볼 수 있다. 따라서 물체들간의 경계점을 조사하여 미리 인식하고 분할 과정에 들어가면 계산량을 훨씬 줄일 수 있다. 구체적 알고리듬은 다음과 같은 recursive 형태로 구현된다.

```
Procedure stripe_seg(posi)
  /*the position data acquired by structured light
  sensor are stored in posi[512]*/
  start=0
  for i=0 to 511
    if posi[i] is on a jump edge
      segment(start, i-1)
      start=i
  end
  segment(start, 512)
end stripe_seg
Procedure segment(i1, i2)
  /*recursive segmenting procedure between
  posi[i1] and posi[i2] position data*/
  find line equation between posi[i1] and posi[i2]
  for i=i1 to i2
    find the distance from posi[i] to the line
  end
  if(max_distance> BOUND)
    segment(i1, max_posi)
    segment(max_posi, i2)
end segment
```

이 과정이 수행되면 range-buffer array, posi[·]에 저장되어 있던 위치값들이 linked list 형태로 바뀐다. 이런 형태로 변환되면 데이타양도 줄뿐더러, 이후에 작업하기도 편리하다.

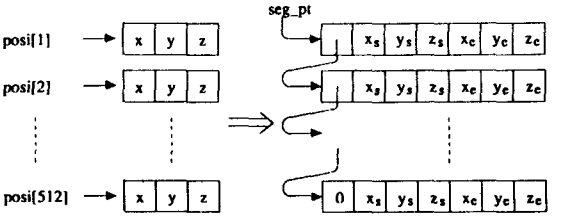



**그림 5.** Stripe fragment 데이타 구조
**Fig. 5.** Data structure of stripe fragment.

## V. 주행 경로 결정

### 1. 물체 인식과 금지 구역(Forbidden Area) 설정

본 논문에서는 이동로보트 주변 물체들의 지면에서 로보트의 높이까지의 범위에 속하는 부분을 지면으로 투영시켰을 때 물체가 점유하는 지역을 금지 영역(forbidden area)이라 칭하고 이 금지 영역의 경계를 금지 영역도(forbidden area map)라 한다. 금지 영역의 의미는 대상 물체와 이동로보트 어느 한 부분과의 충돌 가능성을 내포한 지역이라는 뜻이다.

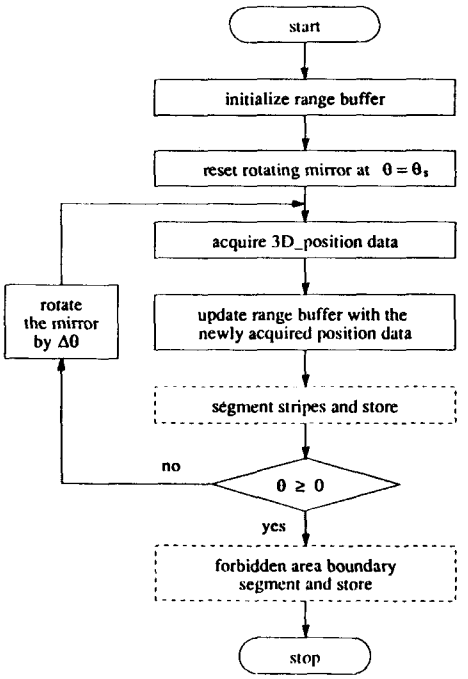



**그림 6.** 주행경로결정 순서도
**Fig. 6.** Flowchart of path determination.

주변 물체와의 거리를 구하고 금지 영역을 설정하는 작업은 그림 6의 순서도로 요약된다. 여기서 reset 각도 $\theta_s$는 이동로보트가 지면을 확인할 수 있는 stripe의 주사각이고, $\theta=0$에서 중지하는 것은 stripe의 주사 장치가 로보트의 상단부에 설치되어 있으므로 로보트가 지면에서 자신의 높이까지의 공간을 인식했음을 의미하기 때문이다. 실시간 주행을 위해서는 순서도내의 점선친 단계를 생략할 수 있다.

### 2. 금지 영역도 저장(Forbidden Area Map Storage)

Range-buffer에 저장된 금지 영역에 관한 평면 정보 역시 카메라 수평 화점 수만큼 sampling된 값이므로 Ⅳ.2절에서 설명한 방식대로 몇개의 선으로 근사화시켜서 저장할 수 있다. 이 때 오차 한계의 크기는 주변 환경의 복잡함에 따라 사용자가 지정한다.

이동로보트가 작업 환경내에서 여러곳을 이동하면서 인식한 금지 영역은 그 순간의 이동로보트의 위치와 방향을 안다면, world 좌표계에 대한 값으로 변환하여 저장할 수 있으며, 최종 global 금지 영역도를 얻는데 이용할 수 있다.

### 3. 경로 결정

이동로보트의 경로 결정은 2단계로 이루어진다. 우선 자신이 이동하려는 쪽의 지면의 존재와 요철의 정도를 확인한다. 건물내부라면 건물바닥을 인식하는 것이다. 그리고 두번째로는 V.1에서 얻은 금지 영역 분포를 연계시켜 실제로 이동가능한 지역을 결정하며 다음 위치와 경로를 선정한다.

1) 건물 바닥의 확인

앞의 V.1에서 설명한 과정에서 선 분할(line segment) 방식으로 임의의 각 $\theta_i \sim \theta_j$사이에 주사된 stripe들을 분할, 저장하는 과정을 함께 수행한다. 저장된 모든 stripe fragment들에 대해 그 높이와 기울기를 확인하여 건물 바닥에 속한다고 인정되는 것들을 서로 연결하면 해당되는 영역을 알아낼 수 있다.

2) 이동 위치와 방향 선정

금지 영역을 선정하고, 그 경계를 선 분할한 후 결정된 segment들로 이동 방향과 위치를 결정한다.

그림 7과 같은 예를 들어 이 과정을 설명하겠다.

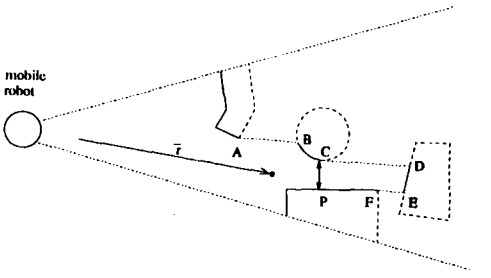



**그림 7.** 주행경로 결정의 예
**Fig. 7.** Example of path determination.

이 과정은 우선 금지영역 경계를 이루는 segment들의 위치를 보아, 그 중에서 불연속 되는 곳, 즉, 자유공간이 존재하는 방향을 찾아낸다.(점 A와 B, C와 D, E와 F) 물론 불연속 되는 곳에 반드시 자유공간이 존재하는 것은 아니나, 현재 위치에서 얻은 정보로는 막혀 있다고 단정할 수 없으므로 이렇게 가정한다.

그리고 공간의 크기(즉 선분 AB, CD, EF의 크기)가 로보트의 직경보다 큼을 확인하면 이동로보트가 통과 가능한 곳으로 일단 선정한다. 그리고 선정된 각 지점에 이르는 건물 바닥의 넓이를 조사해 보고 바닥의 넓이가 로보트가 통과하기에 충분한가를 다시 확인한다. 이 같이 건물 바닥을 확인하는 과정이 필요한 이유는 segment의 불연속 공간의 크기가 로보트보다 크다하더라도 그 곳에 도달하는 경로 주변의 물체들이 로보트의 이동을 제한할 수 있기 때문이다.

그림 7 에서와 같이 segment 불연속 점들의 크기는 CD가 제일 크지만 C와 P사이의 건물 바닥의 넓이가 이동로보트 보다 작으므로 제외되어야 하는 것이다. 위의 조건들을 만족하는, 이동가능한 지역이 여러곳이면 이 중 어느 것을 선택할 것인가는 아래와 같은 간단한 함수 값으로 결정한다.

$$\text{Cost} = K_\theta |\theta_{dev}| + K_d \frac{1}{D_{seg}}$$

여기서

$\theta_{dev}$ = 목표 방향에서의 편이각 (deviation angle from the goal direction)

$K\theta$ = 편이 가중 계수 (deviation weighting constant)

$D_{seg}$ = 불연속한 segment간의 거리

$K_d$ = 거리 가중 계수 (distance weighting constant)

위에서 $\theta_{dev}$는 로보트가 통과 가능한 자유 공간의 위치가 실제 이동로보트의 목표점 방향에서 각 거리상 얼마나 이탈되는가를 나타내는 값이고, $D_{seg}$는 자유 공간(segment가 불연속한 지역)의 크기를 나타내는 값이다. 실제로 로보트의 이동점은 segment 불연속 선상의 중심과 거기에로 연장되는 건물 바닥의 폭을 감안해 결정한다.

본 논문에서는 불연속 segment 구역의 중심까지의 길이만큼을 인접한 바닥의 중심방향으로 진행하게 하였다.(그림 7 에서 벡터r)

## 4. 실험 및 결과

여기서는 이 장에서 제안한 알고리듬의 효용성을 보이기 위하여 생활 주변에 존재하는 의자, 탁상, 휴지통 그리고 앵글등이 배치된 상황속에서 실험을 하였다.

[예 1] (그림 8 참조)

그림 8(a)는 이동로보트의 진행 방향에 의자와 휴지통이 존재하고 전면과 우측면이 벽인 상황이다. 그림 8(b)는 stripe을 $\theta = 24°$에서 로보트의 높이($\theta = 0°$)까지 0.4° 간격으로 주사해서 선 분할 방식으로 저장했던 stripe fragment들을 카메라의 관측점에서 본 것으로 다시 그린 후 실제 물체들과 중첩시킨 것이다. 여기서 실제 물체들의 모서리와 면의 경계가 stripe들이 segment되었음을 알 수 있다.

그림 8(c)는 다시 그려진 stripe fragment들을 서로의 거리와 기울기를 조사해 연결시킨 결과이며, stripe fragment들의 위치와 기울기 값에서 지면에 속하는 것으로 인정되는 영역을 백색으로 채웠다. 이 백색 영역내에서 로보트가 이동해 나갈 경로가 선정

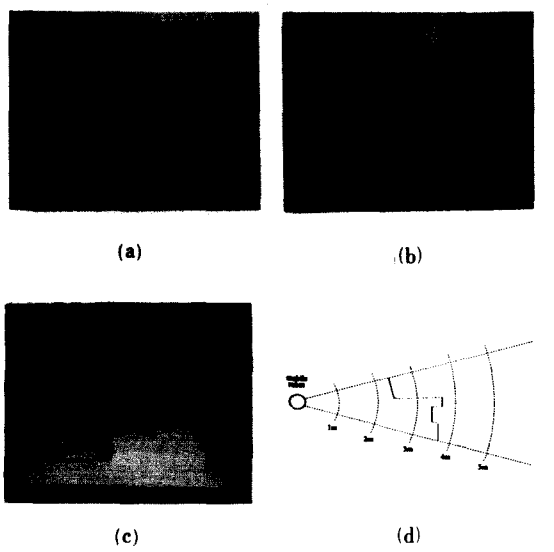



그림 8. 예 1
(a) 원래 image
(b) 저장된 stripe fragment를 원래 image와 중첩시켜 나타낸 결과
(c) 저장된 stripe fragment에서 지면을 인식한 결과
(d) 금지 영역도

Fig. 8. Example 1.
(a) Original image
(b) The result of superposing the original image by stored stripe fragments.
(c) The result of recognizing the stripe fragments belonging to floor.
(d) Forbidden area map.

된다. 이 경우 유의할 점은 의자밑의 바닥도 지면으로 인식되나, 로보트의 이동 경로에서는 제외되어야 한다는 것이다. 실제 이동 경로와 위치는 그림 8(d)와 같이 의자와 벽에 관련된 값들을 지면으로 투영시킨 금지 영역도에서 결정된다.

[예 2] (그림 9 참조)

이 위치는 이동로보트 전면에 TV 받침대가 보이며, 그 사이로 벽등이 보인다. 이러한 상황에서 특히 structured light 센서를 이용해 장애물을 인식하는 방식의 강점이 나타난다. 그림 9(a)와 (b)에서 TV 받침대의 골격과 상자, 밑 받침이 인식되었음을 알 수 있다. 그리고, 그림 9의 (d)에서 전방으로 2m 이상 전진할 수 없음을 알 수 있다.

여기서 stripe의 occlusion 현상을 관찰할 수 있다.

[예 3] (그림10참조)

이 상황은 벽에 대하여 이동로보트가 45° 방향으로 되어 있고, 로보트의 진행방향에 대해 방열기와 탁상이 비스듬히 놓여 있다. 이 경우 방열기를 3차원적으로 저장한 stripe segment들의 분포는 매우 불규칙적이다. 그리고 탁상 중간의 모서리와 벽과 맞붙는 모서리는 stripe에 의해 다시 표현할 경우 완전한 직선을 못이룬다. 달리 말하면 stripe 위치 데이타를 선분할 방식으로 축약 저장하면 불연속 모서리(jump edge)는 잘 모델링되나(예 1과 2 참조) 불연속이지 않은 모서리(non-jump edge) 묘사에는 문제가 있다. 그리고 복잡한 표면 형태를 갖는 물체 묘사에도 문제가 있으나. 이 단점은 선분할 때 오차 한계를 크게 주면 근사화된 하나의 선으로 묘사 가능하다.

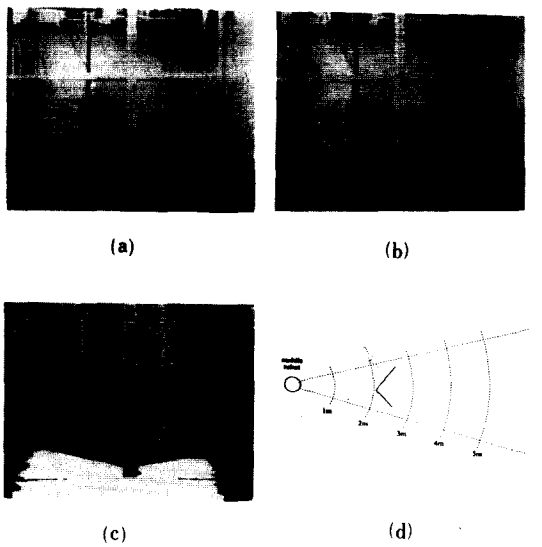



그림 9. 예 2
(a) 원래 image
(b) 저장된 stripe fragment를 원래 image와 중첩시켜 나타낸 결과
(c) 저장된 stripe fragment에서 지면을 인식한 결과
(d) 금지 영역도

Fig. 9. Example 2.
(a) Original image
(b) The result of superposing the original image by stored stripe fragments.
(c) The result of recognizing the stripe fragments belonging to floor.
(d) Forbidden area map.

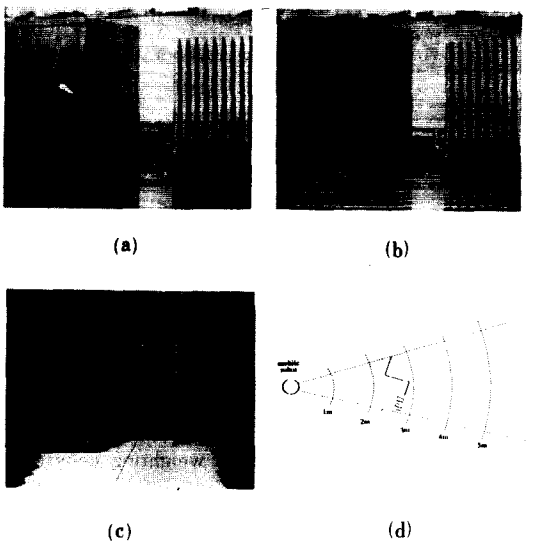



그림10. 예 3
(a) 원래 Image
(b) 저장된 stripe fragment를 원래 image와 중첩시켜 나타낸 결과
(c) 저장된 stripe fragment에서 지면을 인식한 결과
(d) 금지 영역도

Fig. 10. Example 3.
(a) Original
(b) The result of superposing the original image by stored stripe fragments.
(c) The result of recognizing the stripe fragments belonging to floor.
(d) Forbidden area map.

위의 예1~예3과 같은 상황에서 경로결정을 하는데 걸리는 시간은 2분 정도였다. 그런데 이 시간은 stripe이 24°의 영역을 0.4° 간격으로 scan하며, 그 image를 512×512의 카메라로 처리하는데 소요된 것이다. 따라서 카메라의 해상도가 256×256이고 회전간격이 1°라면 걸리는 시간은 10%(12초)로 준다고 할 수 있다.

상황에 따라 가장 크게 변화할 수 있는 변수가 stripe의 회전각인데, 만약 단순한 2차원적인 환경이라면 stripe을 한번만 주사해도 로보트 주행에는 문제가 없다.

## Ⅵ. 주변 환경 모델링

이동로보트에 좀 더 큰 용량의 CPU와 memory를 부착한다면 주행 기능외에 주변 환경에 대한 모델링 기능까지 수행하게 할 수 있다. 그러나, 현실적으로 복잡하고 다양한 3차원 물체들을 이동로보트 자체가 빠른시간에 인식, 모델링할 수 있기를 기대하는 건 아직 무리이므로 일단은 간단한 평면을 대상으로 연구를 진행하였다.

이번 장에서는 선 분할 작업후 저장된 stripe fragment 데이타에서 평면의 존재유무를 확인하고, 존재한다면 그 평면을 규정하는 식과 경계를 찾아내는 과정에 대해 설명한다.

### 1. Stripe fragment 분류

Stripe을 수직 방향으로 이동시키면서 선 분할을 했을 때 축적된 데이타는 아래의 형식을 지닌다. 뒤이어 계속되는 작업은 이 데이타 구조를 유지하면서 진행한다.

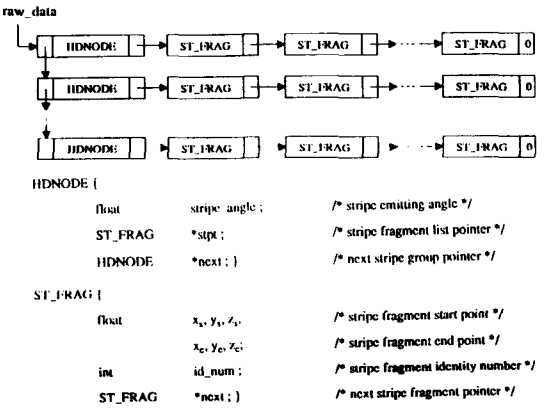



그림11. 원시 데이타 구조
Fig.11. Structure of raw data.

Stripe fragment들을 분류하는 기준은 기울기를 비교하는 방법과 두 segment간의 거리를 이용하는 방법이 있다. Stripe의 fragment의 기울기를 비교하는 방법은 건물 바닥과 같은 수평면을 이루는 stripe fragment을 인식하는데 매우 효과적이며 segment 간의 거리를 비교하는 방법은 동일 물체에 속하는 stripe fragment 집합을 찾아내는데 쓰인다.

### 2. 면에 관한 정보 추출

Stripe fragment들로 구성된 대상 물체들에 관한 데이타 중에 어떠한 평면을 구성하는 군집(group)의 존재 여부를 조사하고 존재한다면 그 평면 방정식을 유도해 내는 과정이다

평면은 다음과 같은 식과, 그 경계면을 규정하는 모서리(edge)로 표현된다.

$$ax+by+cz=d$$

여기서 (a, b, c)는 평면 수직 벡터(plane normal vector)이고, d는 좌표축 원점에서 평면까지의 거리이다.

1) 면 방정식 유도

이 방법은 stripe fragment들 중에 수직 방향으로 인접한 것들(즉 $\triangle\theta$의 차이를 둔 stripe들에서 인식된 데이타) 중에 둘을 선택해 이들의 양 끝점이 어떠한 평면 방정식을 만족하는가를 조사하고, 만족되면 여기서 규정되는 (a, b, c, d)의 평면에 포함될 수 있는 stripe fragment들은 나머지 데이타에서 찾는다. 다만 여기서 같은 평면에 속한다고 인정되는 stripe들을 찾을 때, 그 오차 한계는 잘못된 평면이 등록되지 않도록 되도록 작게 준다. 그리고 같은 평면에 속하는 것으로 분류되는 stripe fragment 들은 수평, 수직 방향으로 서로 인접해야 한다는 조건도 만족해야 한다.

최종적으로 위에서 설정된 평면식을 만족하는 stripe fragment의 갯수가 미리 지정된 숫자를 넘어서면 한개의 seed 평면으로써 등록한다.

2) 평면 경계(plane boundary)의 확장

1)절에서 평면을 구성하는 몇개의 stripe fragment들을 찾았다면 이 평면 경계 주변부에 아직 등록되지 않은 상태로 남아 있는 stripe fragment들이 이 집합에 포함될 수 있는가를 검토한다. 이 경우는 주어진 seed 평면에서 그 영역을 확장하는 의미를 가지므로 1)에서 설정한 오차 한계값 보다 더큰 수치를 준다.

### 3. 평면 정보 저장

일단 어떠한 평면의 존재와 그 식을 얻으면 그 평

면을 구성하는 stripe fragment들을 원래 데이타 구조체의 위치에서 pop하여 함께 연결한다.

이 특정 평면 stripe fragment 집합의 데이타 구조와 공간상의 실제 위치와의 관계는 그림12(a)와 같다.

평면을 규정하는 평면식(*plane-pt)에 속하는 동일 stripe fragment 집합이 형성되면 이들의 위치값 x, y, z에서 페러미터(a, b, c, d)를 least squared error 방법으로 최종 결정한다.

인접한 stripe fragment를 나열하는 식의 그림12(a)의 데이타 구조는 그림12(b)의 형태로 변환된다. 이러한 변환은 stripe의 주사각이 충분히 조밀하다면 별다른 정보 손실없이 이루어질 수 있다.

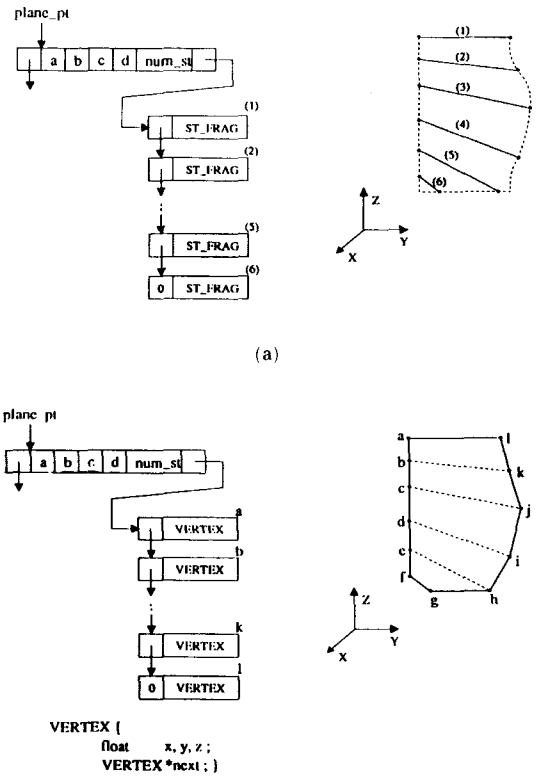



그림12. (a) 평면식에 의해 분류된 stripe fragment 데이타
(b) 평면을 규정하는 형태로 변환된 데이타

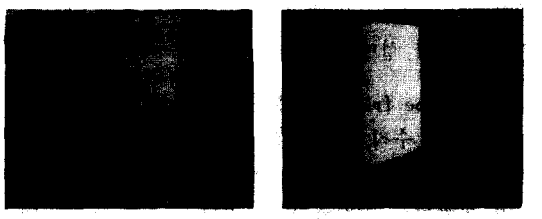
Fig. 12. (a) Stripe fragments group classified by plane equation.
(b) Transformed data which is describing a plane.

예1～예3에서처럼 실제환경에 존재하는 평면의 형태는 그리 복잡하지 않다. 따라서 n개의 stripe fragment 집합에서 2n개의 VERTEX가 얻어지면 이를 선분할(line segment) 방식을 이용해 다시 몇개의 선으로 근사화한다.

### 4. 실험 및 결과

V.4절, 예3의 상황에서 평면을 찾는 알고리듬을 적용해 보았다. 그림13(a)는 로보트 전면의 물체에서 탁상의 양면, 벽, 바닥에 해당되는 4개의 평면을 찾았음을 보여준다. (그림 10(a), (b)참조) 이들 평면의 경계는 동일평면에 속하는 stripe fragment 들의 양 끝점, 즉 VERTEX들을 3cm의 오차한계로 선분할(line segment)한 것이다.

그림 13(b)는 그림13(a)의 결과를 다시 오차한계 5cm로 근사화한 것이다. 이와 같이 오차한계를 키우면 평면의 경계를 규정하기 위한 VERTEX의 수가 줄어든다. 여기서 구성된 평면은 절대좌표값의 오차여부를 떠나, 대상물체의 면을 거의 그대로 복원한 상태이다.



(a) (b)

그림13. 저장된 stripe fragment에서 추출된 평면 정보
(a) 오차 한계 3cm
(b) 오차 한계 5cm

Fig. 13. Plane information extracted from stripe fragments.
(a) Error bound 3cm.
(b) Error bound 5cm.

## Ⅶ. 결 론

본 논문에서는 이동로보트를 위한 structured light 센서의 구현 과정과 원리, 그리고 운용을 위해 개발된 software들에 대하여 설명하였다.

연구를 수행함에 있어 structured light 센서는 쉽

게 조립, setup 시킬 수 있도록 했고, 프로그램 개발에 있어서는 그 수행시간과 필요로 하는 메모리 양의 축소에 관심을 기울였다. 그리 특히 센서를 통해 얻어지는 데이타를 효과적으로 축약 저장할 수 있도록 데이타 변환 작업을 여러번 행하였다. 센서를 통해 얻어지는 데이타의 정확도는 크게 높은 편은 아니나 이동로보트가 진행하면서 계속 물체와의 거리를 궤환(feedback)받는다면 문제가 되지 않는다.

목표로 했던 3차원 환경에서의 주행은 여러 상황에서의 실험을 통해 그 실용성이 확인되었고, personal computer를 이용한 실시간 주행과 3차원 환경인식은 어느 정도 달성되었다고 여긴다.

앞으로 structured light 센싱 방식을 겨냥한 vision board가 함께 개발된다면 수행 속도를 훨씬 높일 수 있을 것이며, 복잡한 형태의 물체인식을 위한 운용 소프트웨어에 대한 연구가 진행되어야 할 것이다.

## 參 考 文 獻


[1] Y. Shirai, "Recognition of polyhedrons with a range finder," *Pattern Recog.*, vol. 4, pp. 243-250, 1972.

[2] M. Oshima, and Y. Shirai, "Object recognition using three dimensional information," *IEEE Trans. PAMI*, pp. 353-361, 1983.

[3] R.J. Poppleston, C.M. Brown, A.P. Ambler, and G.F. Crawford, "Forming models of plane and cylinder faceted bodies from light stripes," *Proc. Int. Joint Conf. Artificial Intell.*, pp. 629-640, 1973.

[4] F. Rocker and A. Kiessling, "Methods for analyzing three dimensional scenes," *Proc. 4th Int. Joint Conf. Artificial Intell.*, pp. 669-673, 1975.

[5] G.J. Agin, "Calibration and use of a light stripe range sensor mounted on the hand of a robot," *IEEE Int. Conf. on R&A*, pp. 660-685, 1985.

[6] C.D. Brown, "Scene segmentation for autonomous robotic navigation using sequential laser projected structured light," SPIE, pp. 145-153, 1987.

[7] J.J. Le Moigne, and A.M. Waxman, "Structured light patterns for robot mobility," *IEEE Journal* R&A, pp. 541-548, Oct. 1988.

[8] D.H. Ballard, "Strip trees: A hierarchical representation for curves," ACM, pp. 310-322, May 1981.

[9] K.S. Fu, R.C. Gonzalez, and C.S.G. Lee, *Robotics: Control, Sensing, Vision, and Intelligence*, McGraw-Hill, 1987.

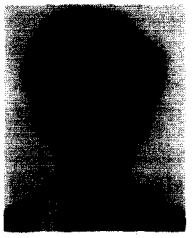
[10] D.H. Ballard, and C.M. Brown, *Computer Vision*, Prentice-Hall, 1982.

[11] R.O. Duda, and P.E. Hart, *Pattern classification and Scene Analysis*, A Wiley-Interscience Publication, 1973.

[12] 이철화, 초음파 센서를 이용한 이동로보트의 항법에 관한 연구, KAIST 석사학위논문, 1988.

[13] 조택일, 3차원 물체의 Range Data 습득 및 재현, KAIST 석사학위논문, 1988. *


## 著 者 紹 介

**李 碩 埈**(正會員)
1964年 4月 12日生. 1987年 2月 서울대학교 전자공학과 졸업. 공학사 학위 취득. 1989年 2월 한국과학기술원 전기 및 전자공학과졸업. 공학석사 학위 취득. 1989年 3月~현재 금성산전연구소 연구원. 주관심분야는 로보틱스, 컴퓨터비젼 등임.

**鄭 明 振** (正會員) 第25卷 第8號 參照
현재 한국과학기술원 전기 및 전자공학과 부교수