대한산업공학회/한국경영과학회 2000 춘계공동학술대회 논문집. 2000년 4월 21-22일. 경남대학교
Session PB3.2

# 차량뒷바퀴윤곽을 이용한 근거리 전방차량인식
# Detection of a Close Leading Vehicle using the Contour of the Vehicle's Rear Tire

노 광 현 · 한 민 홍
고려대학교 산업공학과 첨단차량연구실

**Abstract**

본 논문에서는 차량의 뒷바퀴 윤곽을 이용하여 근거리 전방차량을 인식하는 방법에 대해 설명한다. 차량 앞범퍼 중앙에 장착된 흑백 CCD 카메라에서 입력되는 영상을 분석하면 전방차량의 뒷바퀴 바깥쪽 윤곽이 수직성분을 띄며 도로면과 명확하게 구분됨을 알 수 있다. 이 특징을 이용하여 일정거리 이내에 존재하는 전방차량을 탐지할 수 있다. 또한 탐지된 차량까지의 대략적인 거리를 계산할 수 있고, 브레이크등의 점등 여부를 판단할 수 있다. 이 방법은 차량간 간격이 좁고 저속으로 주행하는 도심지 도로에서 가다서다를 반복하는 Stop-and-Go 차량을 구현하는데 적용될 수 있을 것이다.

## 1. 서론

비디오 센서는 고해상도의 영상 데이터를 제공할 뿐만 아니라 시스템 구성시 작은 공간을 필요로하고 저가로 구현이 가능하므로 주변환경 인식 관련 분야에서 폭넓게 사용되고 있다. 특히, 최근에는 자동차의 안전 주행을 위한 장애물 인식을 위해 널리 사용되고 있다. 도로상의 장애물에는 여러 가지가 있지만 대부분의 경우 차량이 이에 해당하므로 많은 연구가 전방차량탐지 문제로 국한하고 있다.

현재까지 비젼시스템을 이용하여 전방차량을 탐지하기 위한 여러 가지 기법들이 연구되어 왔다. Zielke[1]와 Kuehnle[2]은 전방차량 후면 모습이 차량 중앙을 기준으로 대칭성을 띄는 특징을 이용하였고, Burie[3]와 Broggi[4]는 스테레오비젼을 사용하였으며, Giachetti[5]는 광류흐름(Optical Flow)방법을 이용하였다. 또한 차량에지를 히스토그램화하여 차량의 윤곽을 추정하는 방법도 연구된 바 있다.

기존의 방법들은 주로 원거리에 위치한 차량을 인식대상으로 하였으며, 근거리 차량을 인식하더라도 정확한 거리 산출 및 전방차량의 브레이크 사용 여부를 판단하는 기능에 대해서는 연구된 바는 없다. 하지만 최근 자율주행차량을 도심지 도로에서 적용하려하는 Stop-and-Go 시스템 구현을 위해서는 근거리 차량을 인식대상으로 하며 신뢰할만한 거리 계산 및 전방차량의 감속 정보를 제공하는 브레이크등의 점등 여부를 판단할 수 있는 방법이 필요하다.

따라서 도심지 주행과 같이 전방에 다수의 차량이 근거리에 존재하는 상황에서는 전방차량의 종류도 매우 다양하고 윤곽선을 추출해도 매우 복잡하여 정확한 차량인식이 어렵기 때문에 본 연구에서는 주간 상황에 차량 뒷바퀴의 바깥쪽 경계선이 수직성분을 띄며 도로면과 명확한 밝기차이를 보이는 특징을 사용하여 근거리 전방차량 인식, 거리계산 및 브레이크등의 점등 여부를 판단할 수 있는 방법을 제안한다.

## 2. 전방차량탐지

전방차량 탐지 및 추적을 위해 이용한 차량특징정보는 전방차량의 뒷바퀴 경계선이다. 비젼시스템을 이용하여 전방차량을 탐지하는 경우 대부분의 연구에서는 카메라를 차량내 백미러나 차량 지붕에 설치하였다. 하지만 본 연구에서는 전방차량의 뒷바퀴 바깥쪽 경계선이 관심대상이기 때문에 흑백 CCD카메라를 <그림 1>와 같이 차량 앞범퍼 중앙에 설치하였다. <그림 2>는 차량 앞범퍼에 장착된 카메라로 입력된 전방차량의 모습이며 차량 뒷바퀴의 바깥쪽 경계가 도로면에 대해 수직에 가깝게 나타나며 도로면과 gray-level 값의 차이가 분명함을 알 수 있다.

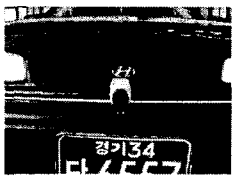


그림 1. 카메라위치.


그림 2. 입력 영상.

이 방법은 모노비젼을 기반으로하고 차량의 특징정보를 이용하여 전방차량을 인식하는 다른 방법들에 비해 차량의 폭과 거리를 정확하게 측정할 수 있다는 장점이 있다. 대부분의 차량특징정보들은 차종에 따라 그 위치가 일정치 않으므로 거리계산에 있어 일반화되기 힘들지만 차량바퀴의 경우는 두 차량이 동일 경사를 갖는 도로상에 존재한다면 훨씬 정확한 거리를 계산할 수 있다.

### 2.1 수직에지성분 추출

차량 뒷바퀴 윤곽은 수직성분을 갖기 때문에 에지를 추출하는데 Sobel 오퍼레이터의 수직에지추출 마스크만을 사용하여 에지 추출 시간을 단축한다. 영상내의 x축과 y축 좌표가 각각 i, j인 픽셀의 gray-level 값을 $f(i,j)$로 정의할 때, 3×3 크기를 갖는 수평에지추출 마스크를 적용한 계산식은 식

(1)과 같다. $\Delta_x f(i,j)$은 픽셀 $(i,j)$을 기준으로 x축방향으로 픽셀값 변화량을 의미한다.

$$\Delta_x f(i,j) = [f(i-1,j-1)+2f(i-1,j)+f(i-1,j+1)] - [f(i+1,j-1)+2f(i+1,j)+f(i+1,j+1)] \quad (1)$$

에지추출은 차량이 존재하는 일부 영역에 대해서만 수행한다. 추출된 에지는 식 (2)를 적용하여 gray-level 변화값의 부호에 따라 상승에지와 하강에지로 구분한다. $g(i,j)$는 $\Delta_x f(i,j)$의 결과값에 에지 문턱값(threshold)을 기준으로 상승에지(rising edge), 하강에지(falling edge), 에지 없음(no edge)으로 분류한다.

$$g(i,j) = \begin{cases} k_1 (rising\ edge) & \text{if } \Delta_x f(i,j) \geq E \\ k_2 (falling\ edge) & \text{if } \Delta_x f(i,j) \leq -E \\ k_3 (no\ edge) & otherwise \end{cases} \quad (2)$$

$k_1, k_2, k_3$ *is constant* and $E$ *is edge threshold.*

전방차량 뒷바퀴 경계선의 gray-level값을 살펴보면 왼쪽바퀴경계에서는 $\Delta_x f(i,j)$이 큰 음수값을 갖는 하강에지가 발생하고, 오른쪽 바퀴 경계에서는 $\Delta_x f(i,j)$이 큰 양수값을 갖는 상승에지가 발생한다. 한 쌍의 뒷바퀴를 찾기 위해서는 하강에지로 시작하여 상승에지로 끝나는 선분들의 모임으로 찾아낼 수 있다.

<그림 3>의 입력영상에 식 (1)과 (2)를 수행한 결과가 <그림 4>와 같이 나타난다. 식 (2)에서 $k_1 = 128$, $k_2 = 255$, $k_3$로 0설정하였다. 흰색 픽셀은 하강에지, 회색 픽셀은 상승에지, 검은색 픽셀은 에지가 없음을 의미한다.

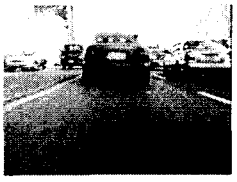


그림 3. 입력영상.

그림 4. 에지추출결과.

### 2.2 후보 선분 추출 및 군집화

전방차량의 뒷바퀴 경계선은 <그림 4>와 같이 하강에지(흰색)로 시작해서 상승에지(회색)로 끝나는 선분을 이루는 한 쌍의 에지가 수직으로 일정개수 이상 나타난다. 하지만 비슷한 특징을 갖는 다른 선분들이 존재하기 때문에 이들로부터 바퀴 경계선만을 찾아내는 작업이 필요하다.

이를 위해서 우선 탐색영역 전체에서 나타나는 모든 선분을 추출하여 $L_i (i=1,\ldots,n)$정의하고 각 선분에 대한 중심점 x좌표 $X_{L_i}$, y좌표 $Y_{L_i}$과 선분의 길이 $W_{L_i}$를 구한다. 이 과정을 통해 많은 선분들이 얻어지는데 FOE(Focus of Expansion) 개념을 도입하여 일부 선분들을 탈락시키다. 전방도로의 차선을 보면 한 점을 향하고 있는데 이를 FOE라 한다. 이 점은 차선의 연속으로 찾아낼 수 있고 모든 차량은 차선 안에 위치하므로 많은 선분들 중에서 FOE 기울기에 맞춰 각각의 영상 y축에서 차선의 폭보다 작아야하고 또한 일정폭보다는 커야하는 조건을 만족해야 한다.

다음 단계에서는 FOE 조건을 통과한 선분 $L_i$를 클러스터링한다. 우선, 각 선분의 3가지 성분을 3차원에 맵핑시킨다. $X_{L_i}$를 x축으로 $Y_{L_i}$를 y축으로, $W_{L_i}$를 z축으로 하여 각각의 $L_i$들을 맵핑 시킨 후 클러스터링 기법을 사용하여 분류하면 m개의 클러스터 $C_i (i=1,\ldots,m)$얻게 된다. 이 중에서 클러스터의 크기가 일정크기 이하일때는 탈락시킨다.

선분들을 3차원상에 맵핑 시킬 때 x축과 z축은 실제 픽셀값을 그룹화하여 범위를 줄이고, y축에 해당하는 점의 개수에 따라 클러스터의 크기에 변화가 심하게 발생하기 때문에 y축의 값은 그대로 사용한다.

3차원상에 매핑된 점들을 클러스터링하기 위하여 본 연구에서는 nearest neighbor method[6]를 사용하였다. $C_i$, $C_j$를 클러스터라고 정의하면 두 클러스터간의 거리는 식 (3)에 의해 정의된다. 클러스터간의 거리는 각 클러스터에 포함되있는 모든 벡터점들 중 가장 짧은 거리에 해당하고, 두 벡터점간의 거리를 구하기 위해 식 (4)를 적용하는데 이 식은 n차원상에 위치하는 두 벡터 $\boldsymbol{a}$와 $\boldsymbol{b}$ 각 요소들간의 절대거리(absolute difference)합으로 정의되는 city block distance[6]를 변형하여 각 차원에 $k_i$라는 가중치를 적용하여 가중화된 거리합으로 정의하였다.

$$D(C_i, C_j) = \min_{\boldsymbol{a}\in C_i,\ \boldsymbol{b}\in C_j} d(\boldsymbol{a},\boldsymbol{b}) \quad (3)$$

$$d(\boldsymbol{a},\boldsymbol{b}) = |a_x - b_x|^{k_1} + k_2|a_y - b_y| + k_3|a_w - b_w| \quad (4)$$

클러스터 벡터점들이 3차원상에 존재하므로 n은 3에 해당하고, 가중치 상수 $k_1$, $k_2$, $k_3$는 각각 x축, y축, z축의 절대거리에 대한 가중치로 크기는 실험을 통해 각각의 값을 결정하여 사용하였다.

### 2.3 전방차량탐지

위의 과정을 통과한 클러스터가 2개이상 발생할 때에는 클러스터의 적합도가 큰쪽을 선택하도록 한다. 이는 가장 가까이 있는 차량을 의미한다. 또한 전방차량의 y축값의 위치에 따라 클러스터의 크기가 달라지는 조건과 차량 바닥이 도로면 색깔에 비해 어두운 색을 띄는 특징을 이용하여 확인 절차를 거치도록 한다. 최종적으로 전방차량이 인식되면 다음 단계부터는 처리영역을 축소시켜 위의 과정을 반복하도록 한다.

추적중인 차량의 위치가 갑작스럽게 변하는 경우를 제거하기 위해서 전방차량의 센터값을 식 (5)와 같이 이동평균기법(Moving Average)을 사용하여 새롭게 결정하였다. $M_x$, $M_y$, $M_w$은 새롭게 탐지된 전방차량의 위치 정보이고, i 인덱스를 갖는 변수는 이전의 차량 위치 정보에 해당하고 i+1 인덱스를 갖는 변수는 이동평균기법에 의해 새롭게 설정되는 차량 위치 정보이다.

$$\begin{aligned} x_{i+1} &= (1-\alpha_1)x_i + \alpha_1 M_x \\ y_{i+1} &= (1-\alpha_2)y_i + \alpha_2 M_y \\ w_{i+1} &= (1-\alpha_3)w_i + \alpha_3 M_w \\ &(0<\alpha_1,\alpha_2,\alpha_3<1) \end{aligned} \quad (5)$$

탐지된 전방차량의 정보가 이전 정보와 비교하여 일정값을 넘어서면 차량인식 에러로 규정하고 새롭게 탐색을 시작한다. 이러한 과정이 일정 시간 이상 반복되면 차량을 탐지할 수 없는 것으로 처리하였다.

전방차량이 탐지된 이후에는 다음 프레임에 차량이 나타날 위치를 고려하여 탐색영역을 유동적으로 변경하도록 한다. 이 작업을 통해 빠른 속도로 전방차량을 추적할 수 있다. 추적하던 차량이 탐지되지 않을 시에는 탐색 영역을 전영역으로 확장하여 재탐색 한다.

## 3. 전방차량의 브레이크 사용 여부 판단

전방차량의 뒷바퀴 수직경계선을 구하고 나면 이 정보를 기준으로 차량의 브레이크등 위치를 예측할 수 있다. 붉은색을 띄는 브레이크등을 인식하기 위해서 컬러비전을 사용할 수도 있지만 흑백영상으로도 주간상황에서는 전방차량의 브레이크등 점등여부를 판단할 수 있다. <그림 5>와 <그림 6>는 동일한 차량이 브레이크를 사용하지 않은 경우와 사용한 경우의 차량후방 모습을 보여주고 있고, <그림 7>은 브레이크등 점등 전후 약 3초동안 브레이크등 영역의 gray-level 평균값을 그래프화 한 것이다.

그림 5.브레이크등OFF.

그림 6.브레이크등 ON.

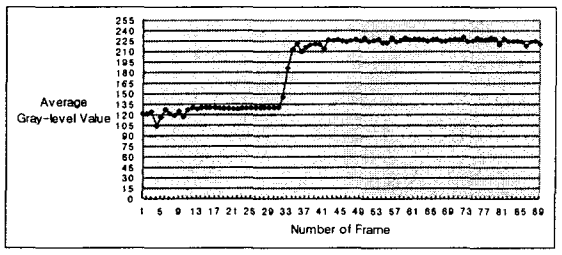



그림 7.브레이크등 점등 전후 gray-level 평균값 변화.

일반 차량의 브레이크등 하우징은 짙은 붉은색으로 되어 있는데 이를 흑백영상으로 처리했을 때에는 <그림 5>에서 볼수 있듯이 상당히 검은색으로 나타난다. 하지만 브레이크등이 켜지면 gray-level값이 흰색에 가깝게 나타남을 볼수 있다. 이 특징을 이용하여 전방차량 브레이크등의 점등 여부를 판단할 수 있다. (브레이크등 탐색방법에 대해서는 생략하도록한다)

## 4. 전방차량까지의 거리 및 방향 산출

컴퓨터 비젼시스템을 이용하여 영상내에 있는 물체까지의 거리를 측정하기 위해서는 일반적으로 스테레오비젼(stereo vision)이 사용되고 모노비젼(mono vision)을 이용해서는 거리를 측정할 수가 없다. 하지만 거리를 측정하고자하는 물체가 수평면으로부터 일정높이에 위치하고 수평면으로부터의 카메라의 높이, 카메라의 기울어진 각도, 초점거리 등의 값을 정확히 측정할 수 있다면 모노비젼을 이용해서 거리를 측정할 수 있다.

본 연구에서는 전방차량과 카메라가 장착된 차량이 동일 경사의 도로상에 위치하고, 차량에 장착된 카메라의 높이는 차량의 진동에 관계없이 일정하다는 가정하에 카메라 한 대를 사용하여 전방차량까지의 거리를 측정하였다.

카메라의 기하학적 모델링을 통하여 전방차량까지의 거리 및 방향을 계산하기 위해서는 좌표계 설정 및 이들 좌표계의 변환과정이 필요하다. 사용되는 좌표계는 모두 직교좌표계로써 화면좌표계 $(X_S, Y_S)$ 카메라좌표계 $(X_E, Y_E, Z_E)$ 월드좌표계 $(X_W, Y_W, Z_W)$들이다. 화면좌표계(screen coordinate)는 CCD 카메라를 통해 입력된 전방차량영상으로부터 차량을 인식하는데 사용되는 좌표계이다. 영상화면 중심을 원점으로하고 수평축을 $X_S$, 수직축을 $Y_S$로 설정한다. 카메라좌표계(camera coordinate)는 카메라의 시축을 $Z_E$로 하고 $X_E$, $Y_E$축은 $Z_E$축을 기준의 왼손법칙에 따라 결정된다. 월드좌표계(world coordinate)는 차량의 위치를 기준으로 설정되는 좌표계로서 차량의 진행방향을 $Y_W$축이라하고, 높이방향을 $Z_W$축, 횡방향을 $X_W$축으로 정의된다. <그림 8>은 설정된 좌표계와 차량간의 위치관계를 보인다.

전방차량의 후방 중심이 화면상의 좌표 $x_s$, $y_s$에 나타나고, 도로상으로부터 카메라의 높이를 h, 카메라의 투사각을 $\theta$, 초점거리를 f라 정의할 때, 전방차량까지의 거리 D와 전방차량과의 편향각 $\beta$를 구하고자 한다.

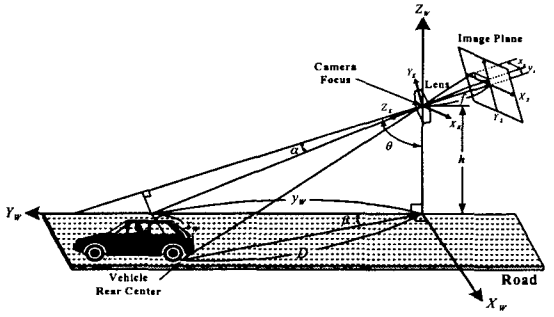



그림 8. 좌표계와 전방차량의 위치 관계.

이를 구하기 위해 화면좌표계의 점 $(x_s, y_s)$은 실제 계산에서는 CCD 셀과의 관계에 따라 실세계 좌표계값으로 변환하여 계산하였다. 전방차량까지의 거리를 계산하기 위해서는 우선 자차량과 전방차량간에 편향된 각도를 계산해야한다.

<그림 8>내의 변수들에 정의 및 계산과정은 다음과 같다.

$f$ : 카메라의 초점거리(*focal length*)

$\theta$ : 카메라좌표계의 $Z_E$ 와 실세계좌표계의 $Z_W$의 각도

$\alpha$ : 카메라좌표계의 $Z_C$ 와 자동차 후방의 중간지점과의 각도

$\beta$ : 전방차량과 자차량의 각도

$D$ : 카메라 초점이 도로면에 투영된 점으로부터 전방차량까지의 거리

$h$ : 도로면에서 카메라까지 높이

식 (6)에서 $\alpha$를 구한 후 이를 이용하여 식 (7)

에서 $y_w$를 구한다.

$$\alpha = \tan^{-1}\frac{y_s}{f} \quad (6)$$

$$y_w = h \cdot \tan(\theta - \alpha) \quad (7)$$

그리고 $\sqrt{f^2+y_s^2} : \sqrt{h^2+y_w^2} = x_s : x_w$에서 $x_w$를 구하면 식 (8)과 같다.

$$x_w = \frac{\sqrt{h^2+y_w^2}\times x_s}{\sqrt{f^2+y_s^2}} \quad (8)$$

최종적으로 구하고자 하는 전방차량까지의 거리 $D$와 전방차량의 각도 $\beta$는 각각 식 (9)과 (10)과 같다.

$$D = \sqrt{x_w^2+y_w^2} \quad (9)$$

$$\beta = \tan^{-1}\frac{x_w}{y_w} \quad (10)$$

## 5. 실험 결과

위에서 제안한 방법을 PentiumⅡ-350, RAM 128MB 기반의 PC하에서 Visual C++로 프로그래밍하였고, 영상처리보드는 Matrox사의 Meteor-Ⅰ을 사용하였으며, 흑백 영상의 해상도는 320×240×256 graylevel을 사용하였다. 실험 결과 입력영상으로부터 위의 모든 과정을 처리하는데 실시간 처리에 가까운 초당 약 29프레임의 처리 속도를 얻을 수 있었다.

아래 그림들은 서울시내 도로 주행상황에서 전방에 나타나는 차량에 대한 인식 결과를 보이고 있다. 인식된 차량에 대해 바퀴 경계선을 기준으로 하여 검은색 사각형으로 나타냈고, 흰색 사각형은 탐지된 차량의 거리에 따라 능동적으로 변하는 탐색 윈도우를 나타낸다. 차량까지의 거리도 차량인식가 동시에 계산되는데 영상내에는 나타내지 않았다. 계산된 거리값은 도로가 평탄한 상태에서는 실제 거리값과 큰 차이를 보이지 않았다. 향후 레이저 거리계를 사용하여 실제 오차를 측정할 예정이다. 브레이크등 인식은 에러율이 커서 좀더 보완이 필요한 상태이다.

<그림 9>는 클러스터링과정 및 탐지결과까지를 보이고 있고 <그림 10>은 탐지된 결과만을 나타내고 있다.

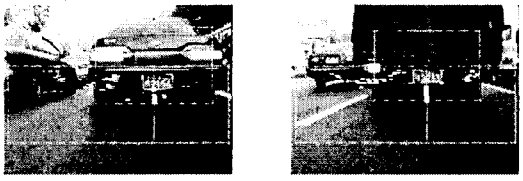
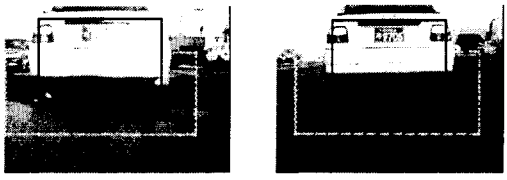


그림 9. 클러스터링 및 탐지결과



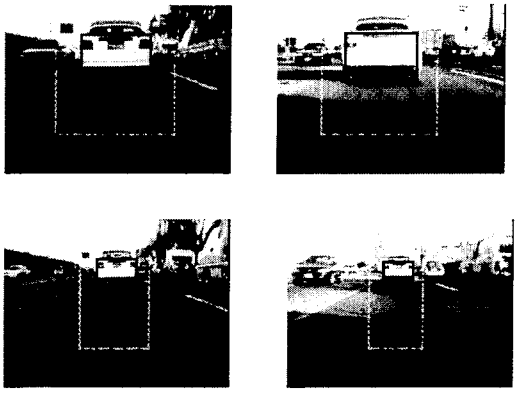


그림 10. 실험 결과 영상.

## 6. 결 론

본 논문에서는 차량 뒷바퀴 경계선을 이용하여 전방차량을 인식하는 방법을 설명하였다. 차량이 위치할 수 있는 영역에서 수직에지성분을 추출한 후 이들을 분석하여 차량의 뒷바퀴 부분을 찾아낸다. 카메라의 기하학적 위치를 이용하여 전방차량까지의 거리를 측정하고 브레이크등의 gray-level값 변화를 분석하여 전방차량의 브레이크 사용여부를 실시간으로 판단할 수 있다. 이 방법은 전방차량이 근거리에 위치할 때 유용하기 때문에 도심지와 같이 주행 중 차량간의 간격이 좁은 경우 전방차량을 인식하고 거리를 계산하는데 효과적일 것이다. 따라서 도심지의 도로에서 가다서다를 반복하는 Stop-and-Go 시스템을 구현하는데 사용할 수 있을 것이며, 저속군집주행에도 적용될 수 있을 것이다.

## 참 고 문 헌

1. T. Zielke, M. Brauckmann and W. von Seelen, "Intensity and Edge-Based Symmetry Detection with an Application to Car-Following", *CVGIP :Image Understanding* 58, pp 177-190, 1993.
2. A. Kuehnle, "Symmetry-Based Recognition of Vehicle Rears", *Pattern Recognition Letters* 12,pp 249-258, 1991.
3. J.-C. Burie and J.-G. Posraire, "Enhancement of the Road Safety with a Stereovision System Based on Linear Cameras", *Itelligent Vehicles* '96 Symposium, September 19-20, Tokyo, Japan, 147-152, 1994.
4. Bertozzi, Massimo, Broggi and Alberto, "Real-Time Lane and Obstacle Detection on the GOLD System", *Intelligent Vehicles*, 1996.
5. A. Giachetti, M. Campani, R. Sanni and A. Succi, "The Recovery of Optical Flow for Intelligent Cruise Control", *Intelligent Vehicles* '95 Symposium, October 24-26, Paris, France, pp 91-96, 1994.
6. Earl Gose and Steve Jost, *Pattern Recognition and Image Analysis*, Prentice Hall, 1996.