

レーンキープアシストシステムにおける 白線認識システムの開発

Development of a Lane mark Recognition System for Lane Keeping Assist System

峯 田 憲 一*
Ken ichi MINETA

鵜 浦 清 純*
Kiyozumi UNOURA

池 田 哲 夫*
Tetsuo IKEDA

要 旨

HONDA Intelligent Driver Support System の主要なコンポーネント、LKAS(Lane Keeping Assist System)のために、カメラで撮影した画像から白線を認識するシステムを開発した。Hough 変換と白線の特徴である正負エッジ点抽出を組み合わせた新しいアルゴリズムにより、ノイズを排除し、処理の大半を専用チップで高速処理することが可能となった。この結果、分岐や二重破線などの複雑な白線をリアルタイムで処理できることが確認できた。

また実画像に対する実験の結果、実用上十分な認識率があり、車線維持制御に利用するのに十分な位置精度を持つ出力が得られることがわかった。

ABSTRACT

A lane mark recognition system that processes images taken by a camera for LKAS (Lane Keeping Assist System) has been developed. This system is a main component of HONDA Intelligent Driver Support System. A new algorithm that was a combination of Hough transformation and extracting positive and negative edges which were feature of lane mark was used. By the algorithm, we can remove noise and execute fast image processing with a special chip. As a result, we can process images in real time even in the complex situation such as a parting road and as double broken line conditions.

In experiments on real images, we can make sure that the rate of recognition is sufficiently high and output data is accurate enough for steering control.

1. まえがき

近年、エレクトロニクス技術の進化に伴い自動車のインテリジェント化の研究開発が盛んに行われるようになった。一部のものはすでに実用化されており、例えば車両前方に向けてレーザレーダを取り付け、前走車を認識して車間距離を調節するシステムが開発されている⁽¹⁾。著者らは、このようなシステムを更に進化させた HONDA Intelligent Driver Support System を開発した。本システムは、カメラで撮影した画像から白線を認識し、その結果を用いて EPS (電動パワーステアリング)のトルクを制御し、ドライバが車線維持のために操舵するのをアシストするシステム、Lane Keeping Assist System(以下 LKAS)を備えている。この LKAS を開発する上で課題となったのは、(1)従来の白線認識システムでは車線維持制御に利用するには位置精度が十分でない、(2)分岐や二重破線など複雑な構造の白線に対応できない、ということであった。そこで我々はこれらの問題に取り組み、LKAS に利用可能な白線認識システムを新たに開発した。本稿ではこの白線認識システムのしくみを解説し、実際の道路画像に対する実験結果について報告する。

2. システム概要

2.1. HONDA Intelligent Driver Support System の概要
HONDA Intelligent Driver Support System は2つの主要なシステムから構成されている。

Fig.1に示すとおり、レーザレーダにより前走車との距離を検出し、アクセルとブレーキを使って車間距離を適切に保つ ACC(Adaptive Cruise Control)システムと、CCD カメラにより車両前方の道路画像を撮像し、画像処理によって白線を検出し、EPS のトルクを制御して、ドライバが車線を維持するのをアシストする LKAS である。本稿で紹介する白線認識システムは、この LKAS のために開発されたものである。そこで次に LKAS の概要を説明する。

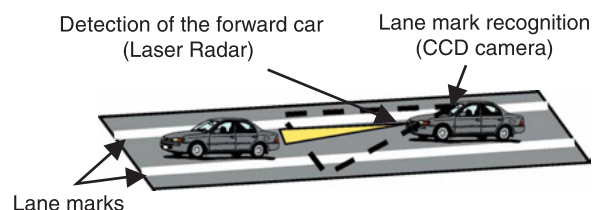


Fig.1 Main functions of HONDA Intelligent Driver Support System

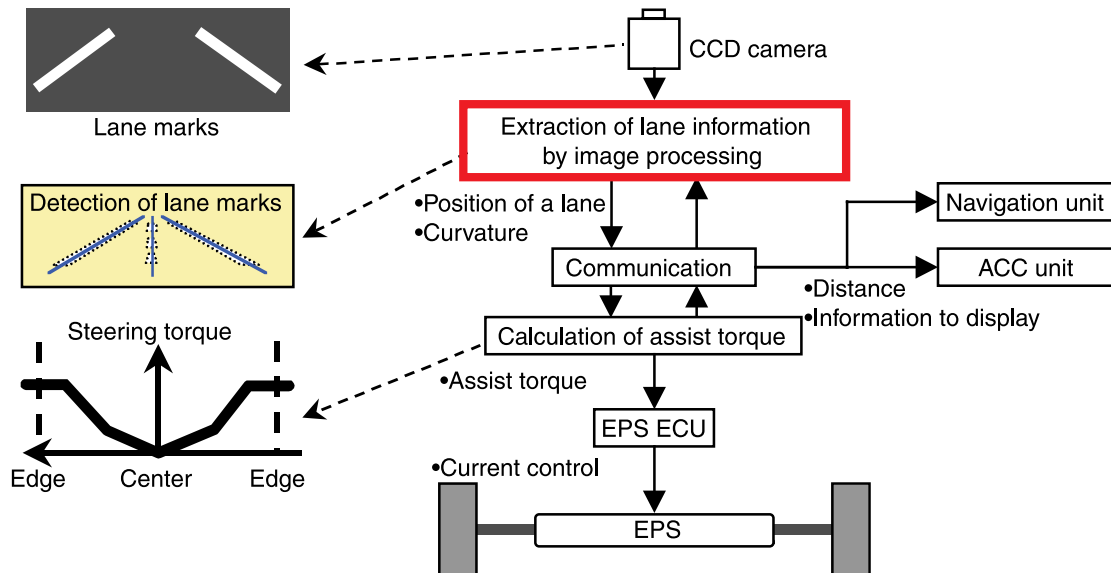


Fig.2 Flowchart of the LKAS process

2.2. LKAS の処理フロー

Fig.2 は LKAS の処理フローの概要である。LKAS ではドライバーが車線を逸脱しないように操舵するのをアシストするため EPS のトルクを制御する。そのために以下の処理を行う。まず、ルームミラー付近に取り付けた CCD カメラにより自車の前方の道路面を撮影する。次にこの画像を画像処理ハードおよび CPU により画像処理し、車線位置と道路曲率を抽出する。その車線情報を用いて車線維持をアシストするために必要な EPS のトルク値を算出する。最後にその指示トルク値を EPS の ECU へ送り EPS を制御する。

Fig.3 はその画像処理の部分の説明したフロー図である。本システムにおける白線認識処理は図に示すとおり、2 つ

のステップに分かれている。第1ステップは白線の候補点列をいくつか検出するステップである。このステップでは白線の可能性がある点列をできるだけ多く検出する。この中には当然ガードレールのような白線ではないものや、分岐の白線のように自車線とは関係ない白線も含まれることがある。しかし、これら対象外の点列の除去は次のステップで行うことにして、ここでは白線を取り損なうことが無いようにすることを第一の目標としている。第2のステップではそれらの候補の中から白線として最も適当なものを選択し、車線位置や道路の曲率などの車線情報を計算する。このステップでは白線の位置をより精度よく求めるために過去のデータを保持している。過去データとの連続性を考慮して候補の中から白線を選択し、さらに過去データと選択した白線点列を合わせて統計的に処理することによって白線位置を推定する。以降ではこの2つのステップについて詳しく説明をしていく。まず3章で白線候補点列の検出手法の説明をし、4章でデータ評価の説明をする。

3. 白線候補点列検出手法

3.1. 対象とする画像

LKAS で処理しなければならない画像の例を Fig.4 に示す。本システムでは画像処理の第一段階として、明るさの違いによる濃度値の変化の影響を無くすためエッジ点処理をする。Fig.4はそのエッジ点処理をした画像である。HONDA Intelligent Driver Support System は高速道路での作動を前提にしているシステムなので、白線認識システムも一般道ではなく高速道路の白線を対象にしている。高速道路の白線で特徴的なのは、インタチェンジやサービスエリアの合流と分岐(b)、比較的きついカーブによくみられる二重破線(f)などがある。ここでいう二重破線とは、通常の白線の内

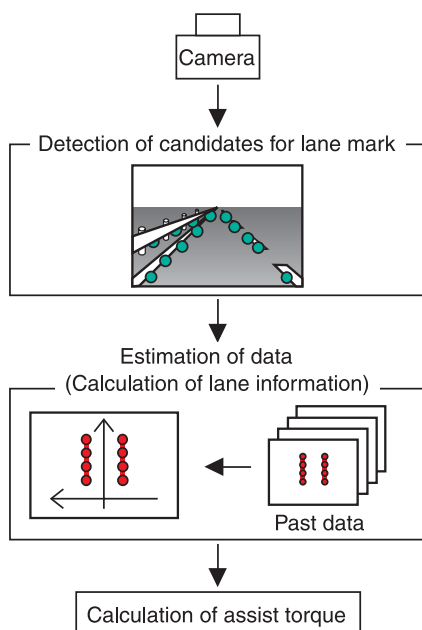


Fig.3 Flowchart of the lane mark recognition

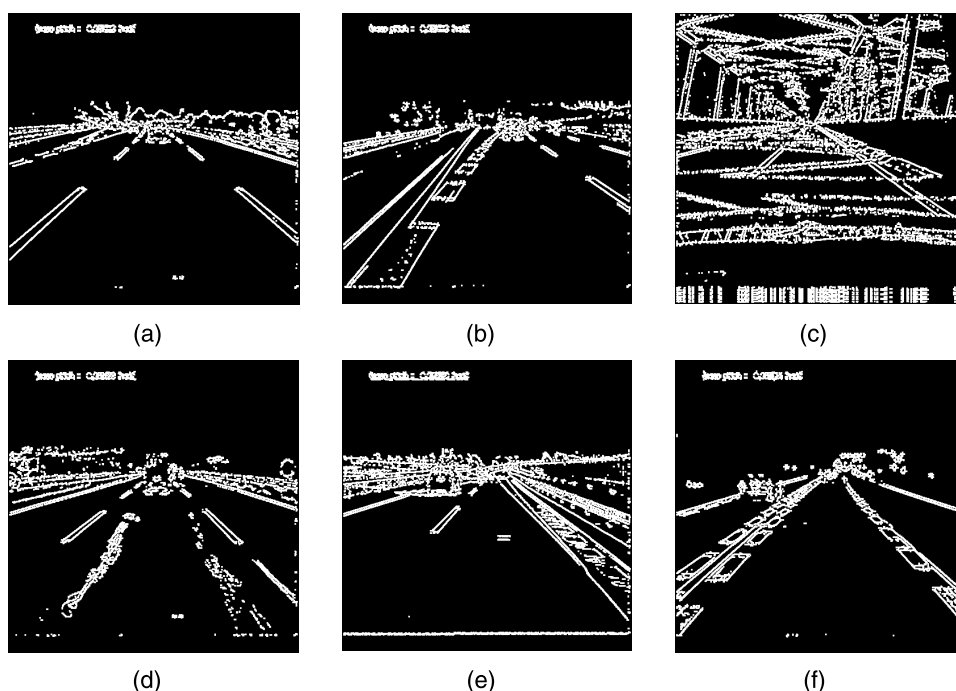


Fig.4 Sample of road images (after edge extracting process)

側にさらに太く短いピッチでペイントされている破線のことで、視覚的にスピード感を増し、ドライバーが減速することを期待するものである。その他に考慮しなければならないものは道路の補修跡 (d) やガードレール、ガードレールによってできる影 (e) などである。さらに複雑な画像の例としては、鉄橋のほりの影 (c) などがある。

3.2. 白線認識の基本原則

画像処理によってどのように白線を認識するのか、その基本原則をこれから説明する。本システムでは白線の特徴として正エッジと負エッジのペアを利用している。正エッジとは画像を左から右に走査していったときに濃度値が増加するエッジ点のことであり、負エッジとは同様に濃度値が減少するエッジ点のことである (Fig.5)。この正・負のエッジ点が次のような条件を満たして配置されている場合、白線を構成する点の候補と考える。

- (1) 所定の距離以内にあること。
- (2) 画面を左から右に走査したとき「正」→「負」の順に配置されていること。

白線を認識する手がかりとしてもうひとつ、エッジ点列

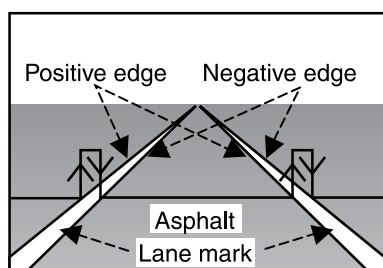


Fig.5 Definition of the positive/negative edge

が直線的にならんでいることも利用する。高速道路においては道路がたとえカーブであっても画面上では比較的直線に近い部分の白線はほぼ直線とみなすことができる。しかもこの部分が画面の大半を占めるので、この手がかりは有効である。画面上に直線的に並んでいる点列を検出する画像処理手法としては、Hough変換⁽²⁾⁽³⁾がよく利用されており、実際にHough変換を利用して白線を認識する研究が行われている⁽⁴⁾⁽⁵⁾。本システムでもこの手法を利用している。以下にHough変換の原理と、Hough変換処理をコンピュータでどのように実現するかを説明していく。まず Fig.6(a)のようなX-Y平面上に点(x, y)があるとする。この点を通る直線Lに対して図のようにρとθをとる。ρは原点からLへおろした垂線の長さであり、θはこの垂線とX軸とのなす角である。このときx, y, ρ, θの間には以下の関係が成り立つ。

$$\rho = x \cdot \cos \theta + y \cdot \sin \theta \quad (1)$$

ここで直線Lを点(x, y)を中心に回転させると (Fig.6(b)), それに伴いρ, θも変化する。その値の変化をρ-θ平面に描くと Fig.6(c) のようになる。次にこの操作をX-Y平面上に直線的に並ぶ複数の点(x_i, y_i) (i = 1, 2, ...) に対して行う場合を示したのが Fig.7 である。このとき、ρ-θ平面上に描かれる曲線の交点はX-Y平面上で各点を通る共通の直線に対応する。よって、点列を通る直線はρ-θ平面上の曲線の交点を求めることによって得られる。

ρ-θ平面の交点を(ρ₀, θ₀)とすると、直線は、

$$x \cdot \cos \theta_0 + y \cdot \sin \theta_0 = \rho_0$$

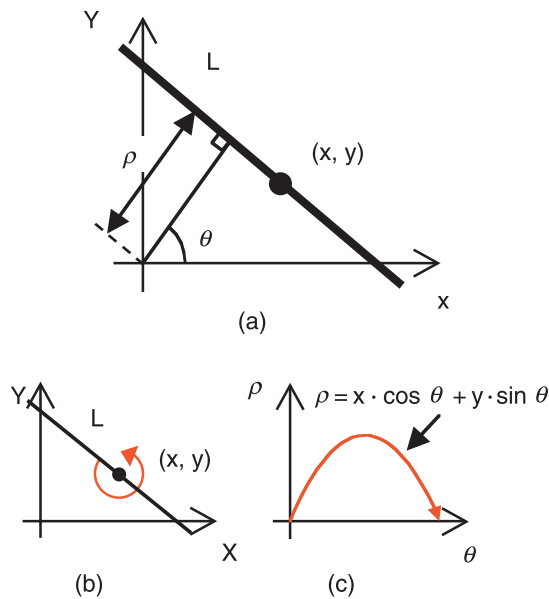


Fig.6 Definition of Hough transformation (1)

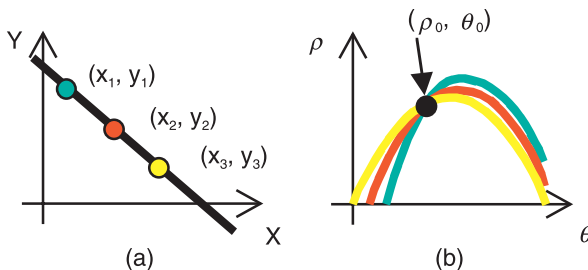


Fig.7 Definition of Hough transformation (2)

と表わすことができる。

次に Hough 変換をコンピュータで実行する手順を示す。

(手順1)

$\rho - \theta$ 平面に対応する2次元配列を用意する (Fig.8)。各要素はカウンタとして機能する。

(手順2)

画像上の点 (x, y) について θ を $0 [\text{deg}]$ から $180 [\text{deg}]$ まで変化させながら ρ を計算し、 $[\rho, \theta]$ で参照される配列要素のカウンタ値を1増やす。(投票処理)

(手順3)

すべてのエッジ点について上記(手順2)の処理を行う。

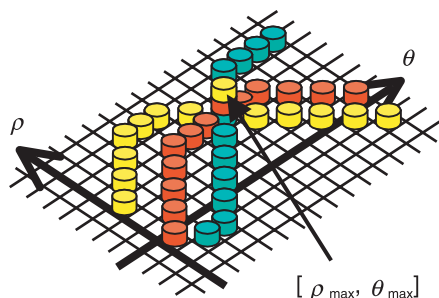


Fig.8 Implementation of Hough transformation

(手順4)

2次元配列のなかでカウンタ値が極大のものを選ぶ。この配列のインデックスを $[\rho_{\max}, \theta_{\max}]$ とすると、 $X - Y$ 平面において、

$$\rho_{\max} = x \cdot \cos \theta_{\max} + y \cdot \sin \theta_{\max}$$

が検出された直線である。

たとえば、 θ の刻みを $1 [\text{deg}]$ に決めて実行したとすると画像上の各点に対して式(1)の計算を180回しなければならない。ただし、処理が単純な計算の繰り返しなので、コンピュータ処理に向いた方法だといえる。

Hough変換について重要な点をまとめると以下ようになる。

- (1)「画像平面 $x - y$ 上の直線点列は Hough 空間 $\rho - \theta$ 上の局所ピークに対応する (Fig.9).」
- (2)「Hough 空間上の局所ピークの高さは対応する直線点列の点数に対応する (Fig.9).」

またその他の特徴をまとめると、Hough 変換の利点はエッジ点列の細かな凹凸にとらわれず、大局的に指向する直線を検出できることである。反対に不利な点としては、例えば二重破線のようにエッジ点が局所的に集中して見られるような場合に、検出したい直線以外の直線が多数検出されてしまうことである。

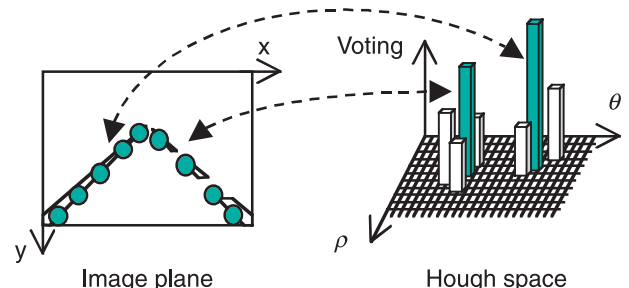


Fig.9 Hough transformation

3.3. HONDA Intelligent Driver Support System 用白線認識手法の特徴

本システムでは、3.2 節で説明した基本原理を組み合わせ、二重破線のような複雑な画像に対応でき、かつノイズの影響を受けにくい白線認識手法を実現した。その特徴を以下に説明する。

最初の特徴は以下のようなものである。

- (1)「二重破線のような複雑な画像に対応している。」

Fig.10 に示すように二重破線の画像に対してエッジ点抽出処理を行うと、エッジ点が非常に密集して検出される。このような場合、Hough 変換を行うと多くの方向の直線成分が検出されてしまい、対象となる白線が埋もれしまう (b)。そこで Hough 変換の際、3.2 節 Fig.5 で説明した白線の特徴を利用することによって余計な方向の直線成分を排除す

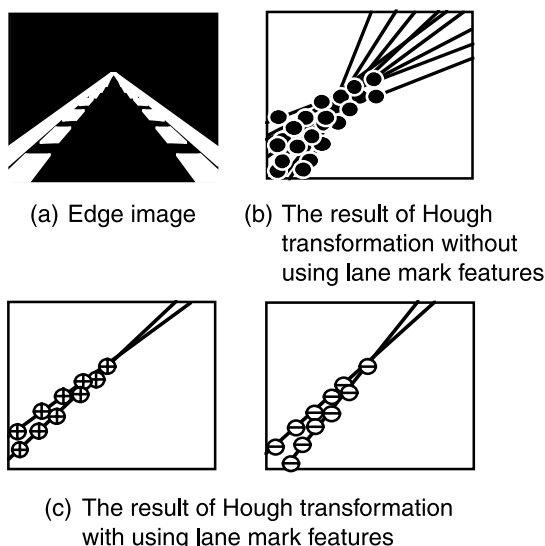


Fig.10 Features of the a lane mark recognition algorithm (1)

る(c)。このため、二重破線のような複雑な白線でも高い確率で認識することができる。

2番目の特徴は以下のとおりである。

(2)「認識結果として直線パラメータではなく点列を出力する。」

Hough変換した結果得られるのはあくまでも直線のパラメータであるが、これをそのまま認識結果として用いると

Fig.11(a)に示すような緩くカーブしているような白線を処理する場合、現実の白線の位置と出力された直線との間に大きなずれが生じてしまい、白線の位置精度が十分出せない(b)。これを回避する方法としては例えば Hough 変換の角度分解能を上げることが考えられる。そうすれば複数の線分で曲線を表現することが可能になり、白線位置を正確に表現することができる。しかし、そのためにHough変換の処理量が数倍に膨れ上がってしまい現実的ではない。そこで、本システムでは白線候補として検出された Hough

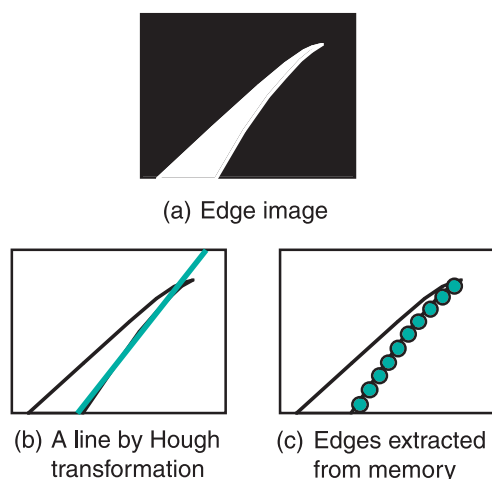


Fig.11 Features of the a lane mark recognition algorithm (2)

ピークそのものを出力するのではなく、Hough ピークを構成している点列を出力するようにした(c)。この出力結果は次のステップであるデータ評価の処理段階でさらに過去のデータと合わせて統計処理され、より精度の高い白線位置が計算される。

3番目の特徴は以下のとおりである。

(3)「ハード化が容易。」

本手法は処理量の大半を Hough 変換が占めている。Hough 変換は、3.2. 節の式(1)を繰り返し計算するだけなのでハード化するのが容易である。実際に本システムではこの処理は専用ハードウェアによって行われており、システムの CPU でソフトウェア処理する場合に比べておよそ100倍程度の高速化を実現している。結果として、限られたハードウェア資源の中で画像全体を処理することが可能となり、分岐など紛らわしい白線があっても複数の候補点列の中から目的の白線を選択できるようになった。

4. データ評価

3章までの処理によりセグメント化された白線候補点列の集合が得られる。この結果を受け、以下の処理を行う。

(1) 真の白線を選別し、検知出来なかった白線を補完する。
(2) 白線の真の位置を推定する。

その処理概要をFig.12を使って説明する。まず各白線セグメントの点列を画像座標系からリアルワールド座標系に射影変換を行う。射影変換は予め静止状態で同定されているパラメータを用いるので、この時点で道路勾配や車両のピッチ、ロール変動等に起因するノイズが載ることになる。次にリアルワールド座標系上で各セグメント間の幾何学的な相互関係を調べグラフによって表現し、同じくグラフによって表現された道路形状モデルとのマッチングを行う。この結果現在走行中の道路形状についての候補集合を得る。この段階で、マッチングをとったそれぞれのモデルについて

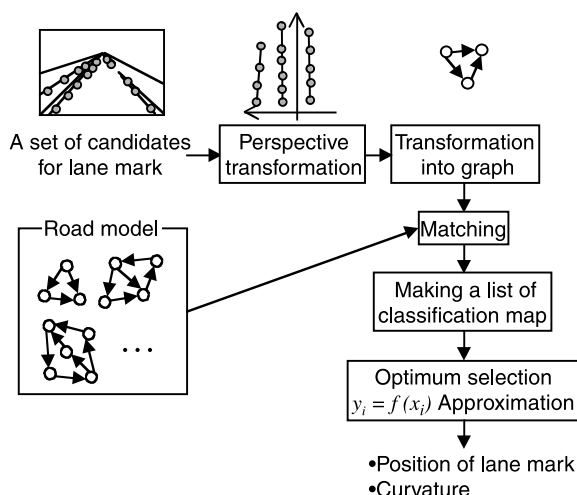


Fig.12 Flowchart of estimating data

て誤検出された白線や欠落している白線を同定し、さらにモデルとの合致度を表す評価値や自車両と各白線との相対的な距離等を算出する。次の段階では、ある短時間の間に走行中の道路構造は変化せず、白線は滑らかに連続しているという前提条件から上記候補のある短時間分集め地図を作製する。そして白線や自車両の車線内位置が滑らかに接続するという条件の下に分類、整理し、現在走行中の道路形状及び地図の候補リストを作る。最後に、この候補リストそれぞれを高速道路の設計基準に照らし合わせ、道幅や曲率変化率の妥当性などを検証し、最適なリストを選択した後、最小二乗近似を用いて白線の真の位置の推定を行う。

4.1. 道路形状の幾何学的な表現

真の白線を選別するために、白線候補の集合と道路形状のモデルとのマッチングをとる。一般にグラフは節点 V と節点間の性質を表す弧 E を用いて、

$$G = (V, E)$$

で表される。 V は属性や属性値 P を持ち、本システムでは以下のように定義する。

$$P = (p, \text{value}), \quad p \in \{\text{実線, 破線, 不明}\}$$

value は p らしさの値

$$E = (e, \text{value}), \quad e \in \{\text{平行, 接続, 無関係}\}$$

value は e らしさの値

相互関係 E は $E_{ij} = E_{ji}$ とすればグラフ G は無向グラフとなり、 $E_{ij} \neq E_{ji}$ とすれば有向グラフとなる。一般にグラフのマッチング問題において、この有向、無向は本質的な

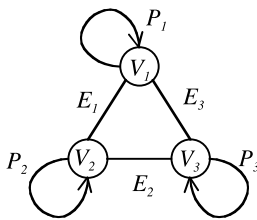


Fig.13 Example of graph

違いはなく、対象とする問題に応じて使い分けられよい。本システムでは「無関係」は無向、「平行関係」は車両の進行方向に応じて有向となるように E を記号化して定義し、鏡像関係にある、つまり幾何学的に相似な構造とマッチしてしまうのを防止し、解の解釈が一意に決定可能なようにしている。

道路形状モデルの例を Fig.14 に示す。このモデルの定義はナビゲーションシステムから得た現在の位置周辺の地図データのリンク情報から構成可能なように設計している。

4.2. マッチングの原理とアルゴリズム

白線候補集合のグラフ G_i とモデルグラフ G_2 とのマッ

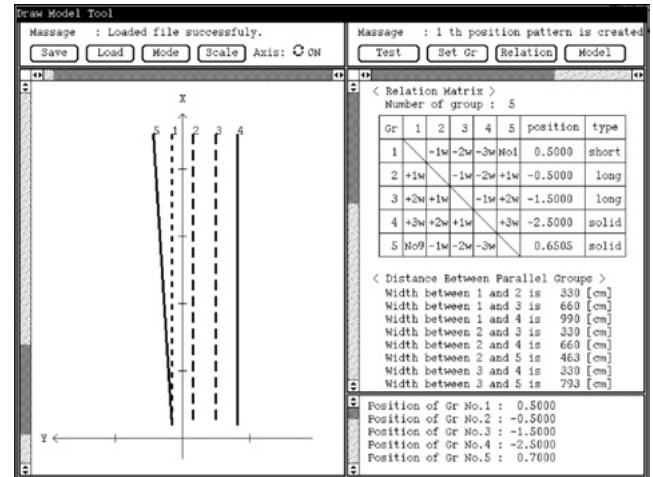


Fig.14 Sample of definition of road model

ングはおおのの部分グラフが同形となるような部分グラフを求める問題として扱うことが出来る。つまり G_p , G_2 の各節点の組み合わせから連合グラフを作成し、 N クリークを抽出する問題に帰着させて解けばよい。連合グラフ自体の作成、及び N クリークの抽出のアルゴリズム⁽³⁾、評価値の算出方法について詳しい説明は省略するが、連合グラフの作成時には車両は道路区分の内側におかれていることを前提とし、車線外となるような V_1 と V_2 の組み合わせを除外する枝狩りを行い、マッチングの高速化を図っている。

4.3. 白線位置と曲率の推定

リアルワールド座標系上での N 本の白線を二次関数 $f = ax_i^2 + bx_i + c_i$ ($i = 1 \sim N$) で表し、係数 a , b , c_i を最小二乗法により算出する。つまり、各関数と元の点列 (x_{ij}, y_{ij}) との誤差を e_i とすると $e = \sum e_i$ を最小にする a , b , c_i を求める。これは以下の連立一次 $N+2$ 元方程式を解けばよい。

$$\begin{pmatrix} \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij}^4 & \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij}^3 & \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij}^2 & \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij} & \sum_{i=1}^N \sum_{j=1}^{n_i} 1 \\ \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij}^3 & \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij}^2 & \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij} & \sum_{i=1}^N \sum_{j=1}^{n_i} 1 & 0 \\ \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij}^2 & \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij} & \sum_{i=1}^N \sum_{j=1}^{n_i} 1 & 0 & 0 \\ \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij} & \sum_{i=1}^N \sum_{j=1}^{n_i} 1 & 0 & 0 & 0 \\ \sum_{i=1}^N \sum_{j=1}^{n_i} 1 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c_1 \\ c_2 \\ \vdots \\ c_N \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij}^2 y_{ij} \\ \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij} y_{ij} \\ \sum_{i=1}^N \sum_{j=1}^{n_i} y_{ij} \\ \sum_{i=1}^N \sum_{j=1}^{n_i} x_{ij} \\ \sum_{i=1}^N \sum_{j=1}^{n_i} y_{ij} \end{pmatrix}$$

ここで各点列 (x_{ij}, y_{ij}) は車両の運動軌跡情報 (x, y, θ) を用いて同時刻での座標系に変換されていなければならない。本システムではこの処理は点列座標を各白線毎の正規行列 $A_i X = B_i$ の係数行列の形に圧縮し、この係数行列毎に時刻補正を行うことで、必要なメモリ容量の削減と座標変換に必要な計算コストの削減をはかっている。Fig.15 に地図リスト中の点列と推定された近似曲線を示す。同図における座標原点は自車両である。車両から離れた白線は

6. 実験結果

6.1. 白線認識率

本システムを実際の道路画像に適用して白線の認識率を調べた。方法は高速道路で撮影したさまざまな状況の画像セットを用意し、机上で処理した結果を次の基準に基づき実験者が判断するというものである。

[基準] 少なくとも前方 50m に渡ってターゲットとなる白線のエッジ点列を検出でき、かつそれが制御に利用するのに十分な数と分布であること。

ただし、画像セットには道路工事の画像や逆光によって白線が見えない場合など、到底白線認識できないと思われる画像は含まれていない。結果は Table 1 のようになった。

Table 1 Result of lane mark recognition

Number of image samples	Number of success	Recognition rate
293	274	94%

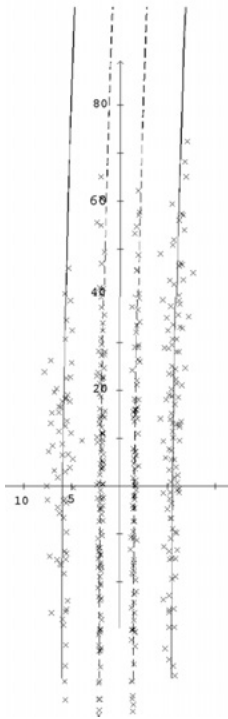


Fig.15 Points in map list and approximation functions

車両運動などのノイズ成分の影響を受けやすくなるが、本推定手法により適切に近似されていることがわかる。

5. ハードウェア構成

Fig.16 に LKAS の ECU の構成を示す。この ECU は画像前処理部、画像 CPU 部、制御 CPU 部の主に 3 つの部分から構成されている。カメラから取り込まれた道路画像はまず、画像前処理部で正負エッジ点検出処理と Hough 変換を専用のハードウェアで行う。これは 3.2. 節で説明した処理である。画像 CPU 部では、画像前処理部で処理した結果を使って、白線候補の絞り込みと白線点列の再構成および 4 章で説明したデータ評価処理をソフトウェアで行う。そして、最終的にこの画像 CPU 部で道路位置情報と道路曲率を算出し制御 CPU へ送信する。

Fig.17 に画像セットの一部に対する処理結果を示す。LKAS はドライバの操舵を前提としており、いかなる場合もドライバの判断を必要とし、ドライバの操作が優先するようシステム対応がなされているため、本画像処理においても安全性を確保しつつ、十分に性能を発揮できるものと考ええる。

6.2. 白線位置精度

実際の高速道路上において車両運動データと同期させてビデオに撮った道路映像を本システムにより机上で処理した結果を Fig.18 に示す。データは画像処理により得られた道路曲率とその分散、および車両の軌跡情報からオフラインで算出した道路曲率である。走行区間は東北自動車道の白河 PA ～那須高原 PA 間のカーブの連続する区間である。同図では時刻 80sec から二重破線の区間となり分散が急激に悪化しているが、同様の状態をテストコース上で再現して実験を行った結果、この程度であればドライバーに違和感のないアシスト制御が可能であることが確認された。

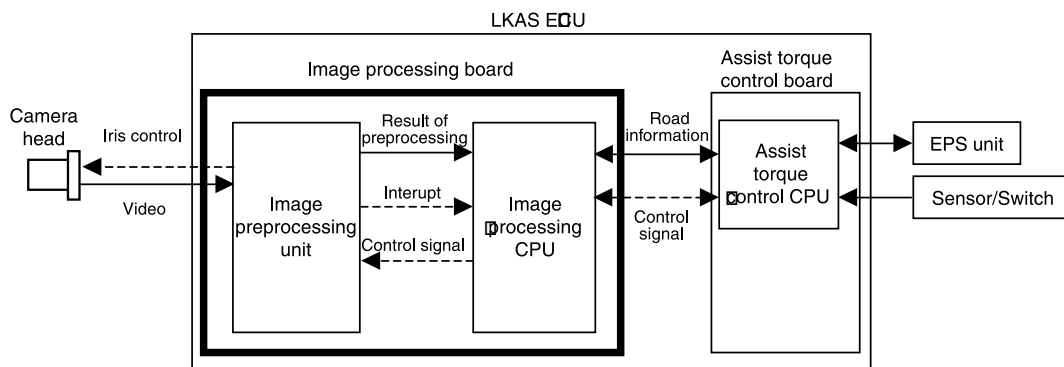


Fig.16 Architecture of LKAS ECU

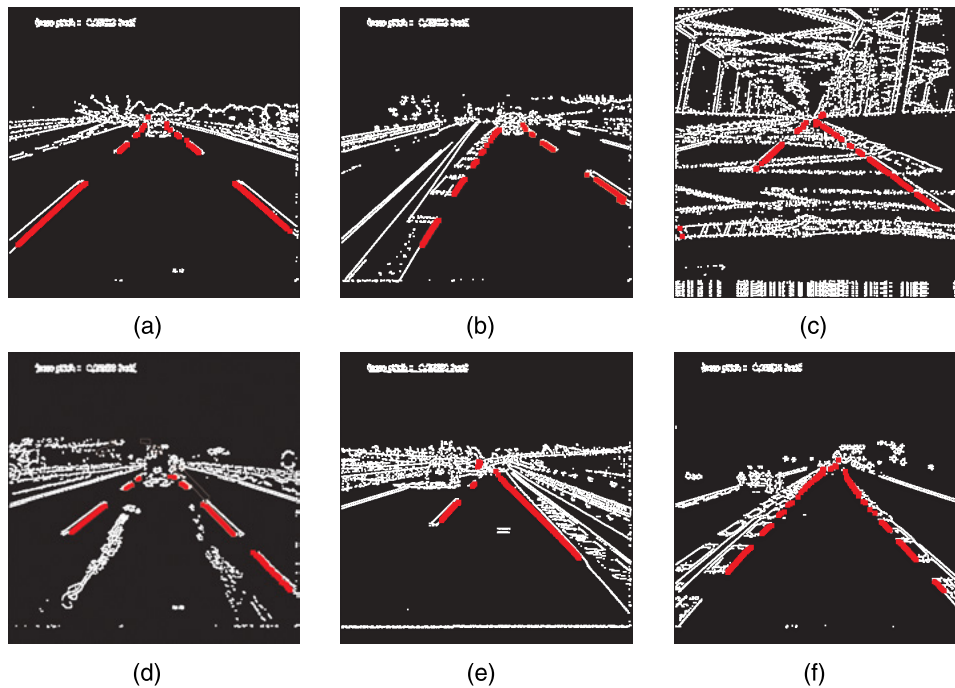
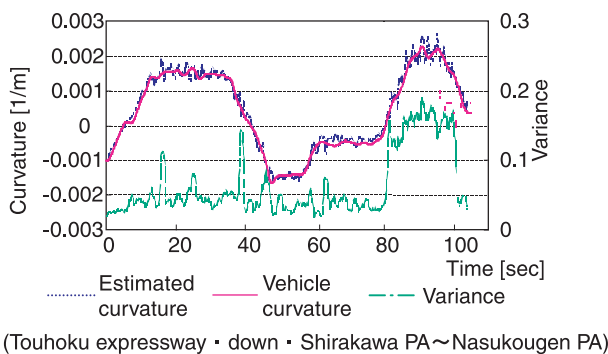


Fig.17 The result of lane mark recognition



(Touhoku expressway · down · Shirakawa PA~Nasukougen PA)

Fig.18 Estimated curvature and variance

7. まとめ

Hough変換と白線の特徴である正負エッジ点抽出を組み合わせた新しい白線認識アルゴリズムの開発により以下の結果を得た。

- (1) 余分な直線成分(ノイズ)を排除し、また単純な計算の繰り返しを専用チップで行うことで処理を高速化して、二重破線や分岐など高速道路における複雑な白線をリアルタイムで認識することが可能な白線認識システムを開発できた。
- (2) 実際の高速道路の画像に対してシステムを試した結果、認識率は実用上十分な高さであることがわかった。また、出力の位置精度は制御に利用するのに十分なレベルであることがわかった。

参考文献

- (1) 藤田泰彦, 菊池隼人, 荻野茂紀, 大崎正滋: インテリジェントハイウェイクルーズコントロール(IHCC)の開発「高速走行時における車速/車間制御システム」, 自動車技術会, 学術講演会前刷集 235 (1999-10)
- (2) Duda, R., et.al: Use of the Hough transformation to detect lines and curves in pictures, Communication of ACM, Vol.15, p.11-15 (1972)
- (3) Ballard, D., Brown, C.: コンピュータ・ビジョン, 4章境界検出, 東京, 日本コンピュータ協会, 1987, p.147-160
- (4) Suzuki, A., et.al.: Lane Recognition System for Guiding of Autonomous Vehicle, IEEE Intelligent Vehicle 92, p.196-201 (1992)
- (5) Hashimoto, K., et.al.: An Image Processing Architecture and a Motion Control Method for an Autonomous Vehicle, IEEE Intelligent Vehicle 92, p.213-218 (1992)

■ 著 者 ■



峯田 憲一



鷺浦 清純



池田 哲夫